BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- BLACK BORDERS
- TEXT CUT OFF AT TOP, BOTTOM OR SIDES
- FADED TEXT
- ILLEGIBLE TEXT
- SKEWED/SLANTED IMAGES
- COLORED PHOTOS
- BLACK OR VERY BLACK AND WHITE DARK PHOTOS
- GRAY SCALE DOCUMENTS

IMAGES ARE BEST AVAILABLE COPY.

As rescanning documents *will not* correct images, please do not report the images to the Image Problem Mailbox.

PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : G06F 12/14 — A1

(11) International Publication Number: WO 97/25675

(43) International Publication Date: 17 July 1997 (17.07.97)

(21) International Application Number: PCT/AU97/00010

(22) International Filing Date: 10 January 1997 (10.01.97)

(30) Priority Data:

PN 7479	10 January 1996 (10.01.96)	AU
PO 0276	6 June 1996 (06.06.96)	AU
PO 0777	1 July 1996 (01.07.96)	AU
PO 1462	6 August 1996 (06.08.96)	AU

(81) Designated States: AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GE, HU, IL, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, TJ, TM, TR, TT, UA, UG, US, UZ, VN, ARIPO patent (KE, LS, MW, SD, SZ, UG), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, ML, MR, NE, SN, TD, TG).

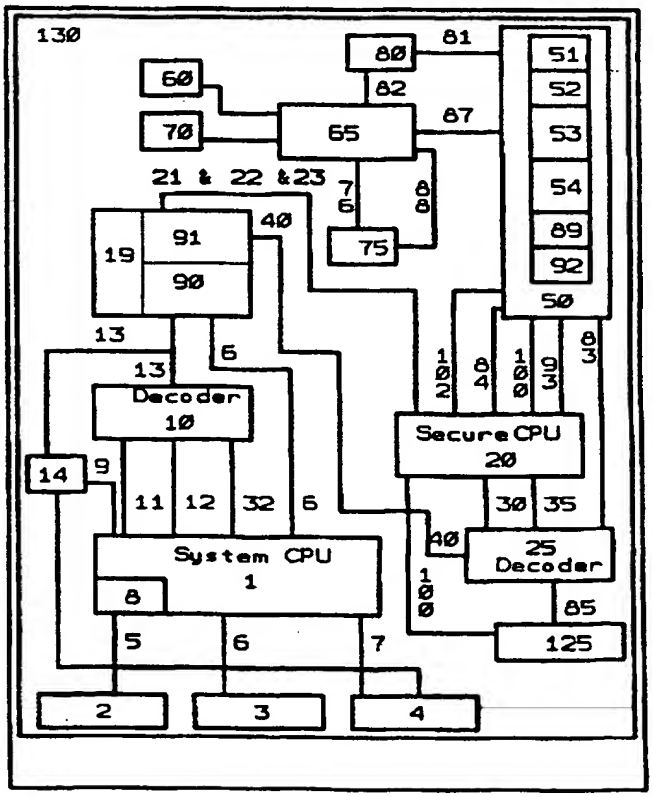
(71)(72) Applicant and Inventor: GRIFFITS, John, Philip [AU/AU]; 298 Savages Road, Brookfield, QLD 4069 (AU).

Published
With international search report.

(54) Title: A SECURE PAY-AS-YOU-USE SYSTEM FOR COMPUTER SOFTWARE

(57) Abstract

A method of renting software that relies on the reversal of encryption processes by the integration of secure processing into the system microprocessor of a user controlled data processing system. It consists of protected software objects, that in addition to being functionally limited to requires reversal of said limitation whithin the system microprocessor, they also have closely integrated information about conditions of use. This is used to distribute computer software on a large scale that may run on any computer. The user is charged on a unit basis. The secure processes described for the system microprocessor will have applications in other secure processes.



FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AM	Armenia	**GB**	United Kingdom	**MW**	Malawi
AT	Austria	**GE**	Georgia	**MX**	Mexico
AU	Australia	**GN**	Guinea	**NE**	Niger
BB	Barbados	**GR**	Greece	**NL**	Netherlands
BE	Belgium	**HU**	Hungary	**NO**	Norway
BF	Burkina Faso	**IE**	Ireland	**NZ**	New Zealand
BG	Bulgaria	**IT**	Italy	**PL**	Poland
BJ	Benin	**JP**	Japan	**PT**	Portugal
BR	Brazil	**KE**	Kenya	**RO**	Romania
BY	Belarus	**KG**	Kyrgystan	**RU**	Russian Federation
CA	Canada	**KP**	Democratic People's Republic of Korea	**SD**	Sudan
CF	Central African Republic			**SE**	Sweden
CG	Congo	**KR**	Republic of Korea	**SG**	Singapore
CH	Switzerland	**KZ**	Kazakhstan	**SI**	Slovenia
CI	Côte d'Ivoire	**LI**	Liechtenstein	**SK**	Slovakia
CM	Cameroon	**LK**	Sri Lanka	**SN**	Senegal
CN	China	**LR**	Liberia	**SZ**	Swaziland
CS	Czechoslovakia	**LT**	Lithuania	**TD**	Chad
CZ	Czech Republic	**LU**	Luxembourg	**TG**	Togo
DE	Germany	**LV**	Latvia	**TJ**	Tajikistan
DK	Denmark	**MC**	Monaco	**TT**	Trinidad and Tobago
EE	Estonia	**MD**	Republic of Moldova	**UA**	Ukraine
ES	Spain	**MG**	Madagascar	**UG**	Uganda
FI	Finland	**ML**	Mali	**US**	United States of America
FR	France	**MN**	Mongolia	**UZ**	Uzbekistan
GA	Gabon	**MR**	Mauritania	**VN**	Viet Nam

TITLE OF INVENTION:

A SECURE PAY-AS-YOU-USE SYSTEM FOR COMPUTER SOFTWARE

TECHNICAL FIELD:

The distribution of software and other information reversibly functionally limited, usually by encryption, requiring reversal by a secure device that may also be used to provide software on a pay-as-you-use basis.

BACKGROUND TO THE INVENTION AND DESCRIPTION OF THE RELATED ART:

The invention describes a method and apparatus that protects software objects. The protected information cannot be used without the assistance of one or multiple secret processing devices. Said *secret processing devices* provide a mechanism for reversing the protection applied to said *information* and said *reversing* may only be activated by certain predetermined secure processes. The process of activating said *reversing* usually ensures that the producer of said *information* and or their agents receive correct payment for usage.

High speed dispersal of information between most computers with access to a modem/telephone line, together with forthcoming means of storing in excess of ten gigabytes of information on a writable optical disk, is likely to lessen the commercial value of information released in clear code format. One clear code copy in the wrong hands could result in its effective worldwide dispersal in a short time.

One objective of the invention is to provide a means of maintaining security applied to information during and after it performs the functions required of it.

The known art describes a means of protecting computer software by requiring the presence of particular devices to operate properly. These devices are secure to varying extents. The problem with computer software is that the protection applied must be reversed prior to providing the information to the system CPU for processing. Once reversed it is accessible to those experienced in the art.

Known art WO 90/13865 describes a process whereby a secure location remote to a potential user supplies an encrypted software object to a user controlled data processing system and a secure method of decrypting said encrypted software object. The software object usually contains information that is continually varying. This provides security by default in that it is a waste of time analysing information that is redundant shortly after its creation. This known art does not provide effective security against objects that, once downloaded and deciphered, may be used in perpetuity as is usually the case with computer programs.

Known art described in AU-A-14856/95 relies on software methods to process the deciphering algorithms used to reverse functional limitations placed on software objects. Said software methods are susceptible to an experienced person generating usable information from protected software objects reliant on this method.

The current invention may be used to significantly strengthen the security and flexibility of the known art described in WO 90/13865 and or AU-A-14856/95. It may also be used as a significantly more secure and flexible replacement for this known art.

Other known art calculates (and this may be by the use of information supplied by an associated computer program) certain values in a secure environment. Said values are passed to an associated computer program and compared with internally generated values. These methods are in effect verifying that said secure environment is present and has presumably been purchased with the computer program. Said secure environment is not providing an essential function absent from said associated computer program, as it is practical to circumvent this protection by disassembly of parts of the program to examine the other side of the equation.

The known art describes a cryptoprocessor (US patents 4465901, 4419079, 4278837, 4168396) that is capable of deciphering instructions and or data in realtime as it is loaded into the central processing unit. Said instructions and or data are usually stored in enciphered format in external memory. This known art is not suitable for use in a user controlled data processing system:

- that may variably have one or multiple programs loaded from a potentially large selection and or said programs may use different decryption parameters; and or
- where the address occupied by a particular program may be different on each occasion it is loaded (said known art is particularly directed at ensuring that an encrypted program will crash with minor variations to its location in the address map); and or
- where one or multiple encrypted programs may need to co-exist with clear code programs in a constantly varying environment; and or
- where it is not usually practical to protect the external memory from tampering and or analysis; and or
- where an interrupt to an encrypted program may direct processing to non-secure methods that may threaten the secrecy of certain information and this may include that within CPU registers at the time of interrupt; and or
- where an encrypted program needs to temporarily transfer processing to an unsecure location; and or
- where an encrypted program needs to protect its stack from analysis; and or
- where an encrypted program exists as multiple modules that are loaded as required and where one or multiple modules may use different decryption parameters that need to be dynamically changed as program execution flows between them; and or
- where different programs in a multitasking environment, that may have different decryption parameters, need to be securely switched on a frequent basis.

The known art describes the programming of software objects into a secure microcontroller. This is restricted to a limited number of predefined functions. However, the known art does not describe the processing of software objects within a user controlled data processing system in conjunction with a secure environmen. that is not practical to analyse, wherein said secure environment (that may be a microprocessor) includes inaccessible information and also provides for external software objects, that may be selected and loaded as required from a potentially large number, to be able to transfer processing (and or pass any required data) to said inaccessible information within said secure

environment, wherein said secure environment includes computer instructions and or data (including that passed) which may be processed in secret within said secure environment to perform important functions and or any other functions that are absent from said software object and that provides for transfer of processing and or data back to said software object as appropriate; and or provide data that is absent from an external software object when appropriately requested by said software object. Said inaccessible information:

- may be preprogrammed into a storage device; and or
- may be greater than the available storage device within said secure environment; and or
- may be dynamically swapped in and out of said secure environment; and or

may be transferred to said secure environment and decrypted within said environment and processed within said secure environment; and this applies for any of the preceding combinations when said secure environment is part of:

- one or multiple system microprocessors, and or
- one or multiple devices attached directly and or indirectly to the user controlled data processing system, and or
- within devices linked via network and or Internet (or equivalent in part or whole).

The known art does not describe any method and apparatus that permits multiple protected software objects, including those protected:

- by software encryption/decryption alone, and or
- by secure decryption within a secret environment, and or
- by secure decryption and secure execution of the ensuing decrypted information within a secret environment,

that allows said multiple protected software objects to concurrently and or otherwise execute in a multitasking and or multiuser and or multiprocessor environment (where said multiprocessors may be the same and or different).

One objective of the present invention is to provide a method and apparatus:

- that overcomes part or all of the aforementioned deficiencies in the known art, and
- that may be used to support a multiplicity of new methods and apparatus for distributing computer software, and
- that may be used to strengthen a number of weaknesses with the current art.

The known art describes a number of methods for distributing software whereby the user pays on 'an as used basis'. These methods include those protected exclusively by software methods. These usually include various software clocks that count down on a predetermined basis, and inactivate the program at the appropriate time. Payment is usually made for the use of a particular object on the terms predetermined. Disadvantage of this method include:

- inherent lack of security;
- the unsecure nature of the protection processes make it unlikely that software vendors will feel comfortable with the process;
- should software vendors make a large selection of software available, users would usually be required to pay for access to the full period predetermined for each program, making it unappealing for users to access a large number of different programs as required (apart from any trial periods);
- lack of flexibility;

- user cannot self determine the amount of time required and pay accordingly.

The security of the process for renting software is improved with known art described in WO 90/13865, wherein there is a secure device within the user controlled data processing system that monitors the time used by a software object downloaded from a service provider. Details of time used is periodically transferred back to the service provider. This method requires the user to be on line to receive said software object and to receive the timing parameters pertaining to said software object. The method also requires the user to remain on line for continued security of the process and to periodically upload elapsed time to the service provider. The user would normally be billed on a predetermined basis for software usage.

The known art does not describe a method and apparatus to provide a secure and secret environment for the secure recording of usage of more than one program at a time in a multitasking and or multiuser and or multiprocessor environment.

The known art does not describe a secure and secret environment that can be securely preprogrammed with a predetermined amount of usage, whereby said usage:

- is prepaid and or
- is a credit limit for use that will be billed at a later date;

and

said predetermined amount of usage remains available for an extended period of time (preferably surviving loss of system power) for use as required, with said predetermined amount of usage appropriately varied according to use of multiple software objects over said extended time, and or

said predetermined amount of usage may be securely updated with additional usage rights as required.

The known art does not describe a secure and secret environment that can:

securely record usage of software objects; and or

securely maintain a record of amounts owing to different vendors and or against different software objects, and or

provide a report on any basis, including usage, and or

temporarily or permanently disable itself in part or whole should said predetermined amount of usage be utilised, and or

temporarily or permanently disable itself should it fail to receive secure confirmation that reports sent to a service provider have been received.

The known art does not describe a method and apparatus to permit a large number of software objects to be created that include information about their particular billing requirements, whereby said software objects are subsequently distributed on a large scale permitting each potential user to use any of the software objects as frequently as they require and only pay for use incurred, said use reducing the amount of usage predetermined within said secure and secret environment. There is no known method and apparatus that compensates for variations between information stored within previously released software objects and that which is current, particularly as it applies to billing information.

It is another objective of the invention to provide a method and apparatus to overcome, in part or whole, the aforementioned deficiencies with the known art, and said method and apparatus may also be used for a number of other described applications. An important objective is the provision of a secure, virtually transparent (to the user) method of renting software for use on a user controlled data processing system (UCDPS), on a usage basis, that in one configuration is independent of any attachment to any devices coupled remotely (eg. telecommunications link) to the UCDPS.

The method and apparatus described to advance the art of protecting and distributing computer software may also be adapted in part or whole to the protection and distribution of other commercially valuable information.

DEFINITIONS:

Replication or **duplication** may be one to many copies and may include replication of part or whole in any combination and or number.

decrypt(ed) and **decipher(ed)** may be used interchangeably and refer to reversal of a previously applied encryption process. Unless relating to a specific decryption process that is a claim of the invention it may be interpreted as being any known method of decryption.

Decode is generally used in the traditional computer sense of decoding addresses etc, however, where the context permits it should be interpreted as for decrypted..

Clear text (or **clear code**) is information that is not encrypted and may be derived from encrypted information and or may have been supplied in as clear code.

Internal to the System CPU (or System Microprocessor) indicates that the hardware and or microcode and or software is on the same integrated circuit substrate; and or that they are on multiple substrates interfacing where necessary using any known method and apparatus within the package of the system CPU; and or part of the device is within the system CPU package and part (or all) external to the System CPU package and attached externally to the System CPU package using any method and apparatus.

A **system CPU** also referenced as **system microprocessor**, is one that a person experienced in the art would consider to be suitable as the primary (or one of multiple primary) processing units in a User Controlled Data Processing System (UCDPS).

Processing or **process** refers to the actual execution of computer instructions and or the manipulation (in any way) of data associated with the computer instructions and or manipulation (in any way) of any other data.

Software Object: A software object is that which a person experienced in the art would consider a software object. Computer programs and or subroutines that constitute part of a computer program are considered software objects. Data pertaining to said computer programs is a software object. Information that is processed by a UCDPS and subsequently displayed as text and or images and or sound for any reason, including as normal output from a computer program and or electronic books (and similar) and or music and or other sound and or visual imagery and or video in the form of motion pictures is a software object.

PCPU: Within this application reference to a PCPU or Protected CPU refers to Secret Processing Device (SPD) embedded within the system microprocessor package of a UCDPS.

ESPD: Reference to an External Secret Processing Device or ESSPD refers to an SPD attached directly or indirectly to any other part of the UCDPS.

End of Definitions.

DESCRIPTION OF THE DRAWINGS:

Figure 1 is a diagram of an apparatus suitable for use as a secret processing device embedded within the system microprocessor.

Figure 2 is a diagram of basic embodiment of an SPD for use external to the system microprocessor.

Figure 3 is a diagram of the address map for secure functions within the system microprocessor.

Figure 4 is a diagram of command port structure.

DESCRIPTION OF THE INVENTION:

A SECURE PAY-AS-YOU-USE SYSTEM FOR COMPUTER SOFTWARE

The invention describes a method and apparatus for the protection of software against piracy and provides a secure process for the mass distribution of software. This is done by functionally limiting a software object and securely linking it with conditions of use and object support information to create a Protected Software Object (or PSO) which must be used with a Secret Processing Device (or SPD) that is directly or indirectly attached to a User Controlled Data Processing System (or UCDPS). This provides a flexible and novel method of using and paying for software. The preferred location of the secret processing device is within the package of the system microprocessor of the User Controlled Data Processing System where the combination is referred to as a Protected CPU (or PCPU). The following describes those aspects considered essential to a full implementation of the invention.

1) a method of distributing software objects from a producer to a potential user comprising the method steps of:

i) providing a secret processing device (or SPD) for direct and or indirect attachment to a UCDPS whereby said SPD is any one or multiple hardware devices that may use any combination of software and or microcode and or any other method to provide a secure and secret environment for processing information and or storing information and that provides the following:

a) any one or multiple methods and or apparatus that:

securely decrypt and execute instructions and or securely decrypt and process data that complies with part or all of the requirements of reversing functional limitations applied using the Oscar method (described later); and or reverses the functional limitations applied using the Groover method (described later); and or reverses any other functional limitations applying to a PSO; and or transfer into the SPD any part of one or multiple PSOs into the SPD that may be necessary to provide any of the functions required by said PSOs; and or access any part of one or multiple PSOs that may be located external to the SPD in order to provide any of the functions required by said PSOs; and or examine the generic and or distinct conditions of use linked to a particular PSO, and or determine a response to said conditions of use; and or respond to said conditions of use;

and or

b) may be embedded, in part or whole, within the package of the system microprocessor of the UCDPS; and or may be within any one or multiple devices attached directly and or indirectly to the system microprocessor and or the UCDPS, and may not disrupt the normal functions of the UCDPS and may in part or whole be used as part of an application that in part or whole is dependent on connection to a distributed data processing system, that may be of any type, including local networks and or intranet (or similar) and or the Internet (or similar), and may benefit from connection to one or multiple remote computers and or any other devices to simplify transmission of various information, however, said secure and secret processing functions, in part or whole, are functional and or remain functional, when said UCDPS that has been provided with said secure and secret processing functions, is used as a standalone unit independently of attachment to remote devices, and said UCDPS may be switched on and off for variable periods of time and or moved to different locations and or reset as frequently as required, without affecting the functions that are provided to said UCDPS;

and or

c) provides an area of secure memory storage devices that is not practical to analyse;

and or

d) provides for partition of secure memory storage devices into one or multiple secure system partitions and one or multiple user partitions whereby programs in system partitions may access user partitions, however, a user partition may not access a system partition unless authorised, and or any particular user partition may not access any other user partition unless authorised;

and or

e) may transfer part or all of protected software objects and or any other software object from unsecure to secure locations for processing and or transfer information from a secure location to an unsecure location; and or

f) may securely decrypt part or all of decrypted parts of protected software objects and or any other encrypted information within said secure locations;

and or

g) may process part or all of one or multiple protected software objects in secrecy, including processing of part or all of that information loaded in encrypted format and decrypted;

and or

h) are programs and or data preprogrammed into the device and or transferred in encrypted format and or in clear code, that assist and or replace any other known software protection and or distribution systems that are dependent in part or whole on user accessible software processes and or unsecure identifying codes to provide protection against unauthorised use of software objects, when part or all of said user accessible software processes and or

unsecure identifying codes are transferred (either by preprogramming and or dynamically as required) to a secure location that permits private processing of the information;
and or
i) have the capacity to detect whether part or all of said suitably configured protected software objects have been tampered with;
and or;
j) may perform secret encryption and or secret decryption in a manner that cannot be analysed, and this may be a software and or hardware function;
and or
k) have the capacity to implement in part or whole, one or multiple hardware devices in programmable logic, preferably programmable logic that may be rapidly erased in the event of tampering, and this includes encryption and or decryption functions implemented in part or whole in hardware, and hardware functions implemented in programmable logic may be dynamically programmed by one or multiple protected software objects;
and or
l) may use any method to determine that there is an attempt to gain access to secret information within the SPD, and said attempt may be physical and or logical analysis, and the response may be any action, using any method, including disabling, temporarily and or permanently, part or all of the SPD and or invalidating in any way part or all of the secret information that may be stored within secure memory storage devices;
and or
m) may securely store information in encrypted and or clear code format in locations inaccessible to unauthorised parties and or securely store information in encrypted format in locations that may be accessible to unauthorised parties, and may detect tampering with stored information;
and or
n) may have the capacity to securely monitor the usage of protected software objects;
and or
o) may securely record the usage of said protected software objects and the record may include a secure breakdown of the usage on a producer and or product and or any other basis, and said record in part or whole is non-volatile;
and or
p) may request and or compel (this may include temporarily of permanently disabling part at least of the SPD) the user of the UCDPS to provide any necessary reports of usage to a service provider and or to any other location;
and or
q) may confirm that said reports have been received as required;
and or
r) does not require modification of the User Controlled Data Processing System operating system;
and or
s) may not require special routines to intercept calls to said system operating system;
and or
t) may identify the type of protected software object and act as required;
and or
u) provides or have access to one or multiple tamperproof, non-volatile source of time and or date;

and or

v) provides or have access to one or multiple tamperproof timers;

and or

w) provides one or multiple methods of identifying at least one tamperproof environment, this may include the use of an electronic signature;

and or

x) provides one or multiple secret codes and or programs that are unique to a particular SPD and or that are common across particular groups of SPDs;

and or

y) provides one or multiple programs, that may be preprogrammed (into the SPD) and or transferred (into the SPD) as required, that use secret information unique to the SPD to decrypt software objects;

and or

z) may process multiple protected software objects in a multitasking environment, this may be transparent to the UCDPS operating system;

and or

aa) include functions, preferably implemented in reprogrammable secure memory, that may be edited and or modified and or deleted and or expanded and or in any other way altered, in a secure manner and usually transparently to the user of the UCDPS, enabling appropriately configured PSO(s) to adapt the secure information in the SPD for any purpose, including: making multiple SPDs identical in part at least (including multiple PCPUs in a multiprocessor system); and or create one or multiple applications not currently available to the SPD; and or that permits any current application to be dynamically adapted, including dynamically reprogramming various hardware functions implemented in part or whole with reprogrammable logic connections; and or dynamically modifying decryption processes;

and or

ab) are programs and or data preprogrammed into the device and or transferred in encrypted format and or in clear code that assist any function described for the correct processing of protected software objects;

and or

ac) include secure memory that stores various internal system routines and may be loaded with externally supplied objects for decryption and or execution and or any other purpose;

and or

ad) may decide to reverse one or multiple functional limitations on one or multiple PSOs based on said conditions of use, where said decide is in part at least autonomous to the SPD and based in part at least, on secure processing internal and or external to the SPD of generic information applicable to multiple PSOs, that may include a plurality of any information states within and or external to the SPD, including one or multiple electronic credits that is modified (directly or indirectly) in response to use of PSOs on time and or events used and or any other basis, and as long as the requirements of one or multiple PSOs and or SPDs are complied with, the user of said UCDPS may be able to execute and or process one or multiple PSOs on the same basis as if they were unprotected software objects;

ii) providing a software object;

iii) modifying part or all of said software object such that it is functionally limited to run on only a UCDPS fitted with a SPD and or equivalent and the functional limitation is by the Oscar method as defined below and or by the Groover method as defined below and or by any other method and said functional limitation may be of one or multiple essential parts of the software object, preferably such that it is not practical to regenerate the original software object from any parts that are not functionally limited, and said modifying is preferably done at a secure location (also referenced as a service provider) that has access to part or all of secret information contained within the SPD and for any particular functionally limited software object the functional limitation may only be reversed on a specific SPD with any unique characteristics necessary to reverse the functional limitation, or the functional limitation may be reversed on a plurality of SPDs characterised by common characteristics necessary to reverse the functional limitation; and or

modifying part or all of said software object, using any method, such that it is securely linked in part or whole, using any method, to one or multiple conditions of use, also referenced as PCPU Inclusion Commands (or PIC), that in part or whole are tamperproof and that include any code that directly or indirectly identifies the producer of the software object and or identifies the software object such that when an SPD interacts with the software object it may record use of that particular software object and or use of PSOs by a particular producer and or use on any other basis, in part or whole, where the record of use in part or whole is used in determining remuneration to the producer and or any other parties; and or the conditions of use include any code that contains information which may be used by the SPD to determine if the software object:

is permitted to execute in part or whole on a units of time used basis, and if permitted, what fee should be applied for the use of the software object and said fee may be any unit of measurement and is preferably a generic units of use basis and said generic units may be attributed any real currency value at any stage;
and or

is permitted to execute in part or whole on an events occurring basis, for example the number of times one or multiple parts of the program are loaded and or executed and or any other measurable events basis, and if permitted, what fee should be applied for the use of the software object and said fee may be any unit of measurement and is preferably a generic units of use basis and said generic units may be attributed any real currency value at any stage;
and or

is permitted to execute on an unlimited basis subject to a fee, and if permitted, what fee should be applied for the use of the software object and said fee may be any unit of measurement and is preferably a generic units of use basis and said generic units may be attributed any real currency value at any stage;
and or

is permitted to execute on any type of limited basis subject to a fee, and if permitted, what fee should be applied for the use of the software object and said fee may be any unit of measurement and is preferably a generic units of use basis and said generic units may be attributed any real currency value at any stage;
and or

requires entry of one or multiple data keys of any type prior to initiating use of part or all of the software object for the first and or any other time on a particular SPD and may include whether or not a fee is to be charged for providing the data key;

and or
requires any other restrictions to be placed on use;
and
any software object modified in part or whole as described is referred to as a Protected Software Object (or PSO);
said Oscar method, is any functional limitation of part or all of a software object by any method of encryption, usually at a secure location remote to the user, where part or all of the reversal of the encrypted information, by decryption and or any other method, occurs within a secure environment directly and or indirectly attached to a UCDPS such that part or all of the instructions and or data of the software object reconstituted by said reversal are not accessible to analysis by any unauthorised party and the execution of part or all of said instructions and or the processing (using any method) of part or all of said data that is not accessible to analysis by an unauthorised party remains in part or whole inaccessible to analysis by any unauthorised party. The result is that part at least of the functional limitation placed on a software object is not compromised by the process of using said software object;
said Groover method is any functional limitation of part or all of a software object by deletion of part or all of the information within the software object, usually at a secure location remote to the user, where part or all of the reversal of the deletion, by any method, occurs within a secure environment directly and or indirectly attached to a UCDPS such that part or all of the instructions and or data of the software object reconstituted by said reversal are not accessible to analysis by any unauthorised party and the execution of part or all of said instructions and or the processing (using any method) of part or all of said data that is not accessible to analysis by an unauthorised party remains in part or whole inaccessible to analysis by any unauthorised party. The result is that part at least of the functional limitation placed on a software object is not compromised by the process of using said software object;

iv) providing one or multiple PSOs onto computer-accessible memory media and or any suitable apparatus for electronically transferring said PSOs to a potential user, and preferably the conditions of use attached to said one or multiple PSOs permit said PSOs to be used on a time or events used basis in a UCDPS suitably equipped with a SPD that has sufficient aforementioned units of measurement stored within and or securely accessible;

v) shipping said one or multiple PSOs on computer-accessible memory media to a potential user and or electronically transferring said one or multiple PSOs;

vi) loading said one or multiple PSOs into a UCDPS and executing as permitted by conditions of use;

vii) where required by the conditions of use or any other reason, a means for the user to:

- request the supply of one or multiple units of measurement that may be required by the SPD for any purpose, and or
- receive one or multiple said units of measurement, preferably in suitably encrypted format, that may use any method, and transfer said units of measurement into the SPD, and or accessible to the SPD, and or
- request the supply of one or multiple data keys that may be required by the SPD, and or
- receive one or multiple data keys and transfer said data keys into the SPD, and or accessible to the SPD, using any method, and or

- generate one or multiple reports of software usage and or any other information that may be required, and supply said reports to service provider and or any other external location, as required, and or
- receive one or multiple codes confirming that said report has been received and supply said one or multiple codes confirming into the SPD and or accessible to the SPD, and or
- request the service provider and or any other authorised party for one or multiple codes that may be used to reactivate part or all of the SPD that may have been disabled for any reason
- receive one or multiple codes to reactivate part or all of the SPD that may have been disabled for any reason and transfer said codes into the SPD, and or accessible to the SPD and

for any of the preceding, the information generated by the UCDPS and or received from the service provider is preferably transferred electronically, however, any other combination of methods may be used including mailing of computer-accessible memory media containing the information.

PREFERRED IMPLEMENTATION OF THE INVENTION:

To assist with understanding the invention, reference will now be made to the accompanying drawings which show one example of the invention. In the drawings, Figure 1 shows an apparatus that is suitable for use as a secret processing device embedded within the system microprocessor.

Throughout this description and the accompanying drawings, many signal lines are represented by a single line and an identifying symbol. This may represent any number of signals, for example, a certain logic function output may clock, clear and set a flip flop, however, usually only one signal line will be shown to represent all three. In the case of various buses, the lines represent whatever number of signals constitute said bus or whatever subset of said bus is relevant for the logic functions it may be entering or leaving. Many control lines are not described or shown in this description as it will be obvious to anyone experienced in the art, where, when, and how, they should be used in order to make functional any apparatus described; descriptions are detailed when needed to help clarify the implementation of any particular function. Throughout this description, the polarity of signals is usually immaterial and not discussed unless of specific consequence; it will be whatever is required in a practical implementation of the invention. When a latch or other device is set or cleared the alternative arrangement is allowed for. While a latch or register is a commonly used storage device in parts of this description, it may be replaced with any other logic and or combination of logic and or software and or microcode that results in a similar outcome.

The invention describes:

1. a method of reversibly functionally limiting a software object that requires a secret processing device (or SPD) to reverse part or all of the functions of the reversible functional limitations and preferably includes a method of securely linking the conditions of use that apply to a particular reversibly functionally limited software object to said reversibly functionally limited software object such that this information may be used in part or whole to determine whether to permit the SPD to reverse the reversibly functionally limited software object. The conditions of use are preferably an integral part of the reversibly functionally limited software object and or supplied as one or multiple other modules that are linked in a manner that prevents the unauthorised separation of conditions of use and reversibly functionally limited software object. This produces a protected software object (or PSO) which may be distributed to a potential user and loaded onto a UCDPS and includes instructions to the SPD on how it may be

distributed to a potential user and loaded onto a UCDPS and includes instructions to the SPD on how it may be used. This permits objects to be widely distributed and used on stand alone UCDPSs conditional on the SPD that is required to reverse, in part at least, the reversible functional limitations, complying with the conditions of use. The conditions of use may also be supplied in any other way, e.g. as separate modules and may be loaded, or otherwise linked, into an SPD transparently to the operating system of the UCDPS or by using said operating system.

When a PSO is securely linked with conditions of use it may be used on a UCDPS equipped with an SPD without any extra intervention by the user than would normally be required for the protected object in its native software object form, with the exception of any requirements that the SPD requires of the user.

2. an apparatus referenced as an SPD that has various secure system functions that allow it to interact correctly with one or multiple reversibly functionally limited software object prepared for use with one or multiple SPDs. The SPD includes an internal secure and secret operating system referred to as secure system functions. They interact in any way required to appropriately reverse in part or whole, reversibly functionally limited software objects. The secure functions of the SPD may have other applications.

The preferred embodiment of an SPD is included within the package of the system microprocessor; such a combination may be referred to as a protected CPU (or PCPU). An SPD may be directly and or indirectly attached to the UCDPS external to the package of the system microprocessor; this is referenced as an ESPD. A PCPU may include multiple system microprocessors. There may be multiple PCPUs within a UCDPS. There may be multiple ESPDs within a UCDPS. Multiple SPDs in any location may interact in any way and combination with any others or not at all. The embodiment of a system microprocessor to implement the apparatus of the invention is predominantly dependent on the use of secure memory storage devices of various types and an ability to securely process information within these devices and a person experienced in the art will be able to arrange logic, software and microcode in many combinations to effect versions of an SPD and PSO that are within the spirit of the invention. This arrangement permits the secure functions required of the present invention to be implemented. A person knowledgable in the art will appreciate that the secure processes used for the invention may have multiple other secure applications. The known art does not describe a system microprocessor suitable for use in a UCDPS that provides the secure processing functions described in this embodiment. The invention allows for any system microprocessor that provides the apparatus and or functions described in the application.

Figure 1 shows a block diagram of a system microprocessor that may communicate with a secure microprocessor that is securely linked to one or multiple secure functions, including secure memory, secure realtime clock and other secure functions. When the secure memory is programmed with appropriate information, the combination of software routines and embedded hardware functions and changes to the microcode of the system microprocessor provides all of the requirements of an SPD securely embedded within the system microprocessor package. This device may be used to replace the existing system microprocessor in a UCDPS and, subject to being supplied with any information required to meet the conditions of use attached to a PSO, may execute that PSO as if it were a normal software object. It will be appreciated by those experienced in the art that there are many ways of combining logic, software and microcode to implement the device as described.

Figure 1 shows the silicon chip 130 of the system microprocessor 1. The system microprocessor 1 normally interfaces with external locations via an address bus 5 and address buffers 2 and data bus 6 and data buffers 3 and various control logic 7 via buffers 4. Buffers 2, 3 and 4 are enabled/disabled during normal processing by system microprocessor 1 via control line 9. Instructions are interpreted and implemented by a combination of microcode and logical devices within the instruction execution block 8, located within system microprocessor 1. The apparatus of the invention needs to communicate with the system microprocessor 1 and this is most readily implemented with dual port memory 19, a memory that allows read and write accesses by two devices to the same addresses on an asynchronous basis. There are many ways of achieving an equivalent result. As described in this embodiment the DP memory 19 is not intended to store secure information; it is functioning as a port between unsecure and secure processes and it is not practical for an unauthorised person to access secure information without very complex codes. The invention allows for the recording of failed attempts at access and may disable itself to prevent repeated attempts to compromise secure elements.

The system microprocessor side of the DP memory 90 may be decoded into the normal address space of the UCDPS, using any known decoding apparatus, however, the preferred method is to make the addresses occupied by the 90 side of the dual port memory 19 a separate address space to the UCDPS. This is done by providing an instruction, referenced as a transparent address activator or TAA, that, depending on the attached opcode, performs a number of functions.

The primary interaction of the system microprocessor 1 to dual port memory 19 will be to read and write data between UCDPS addresses and dual port memory 19 for transfer into secure functions 50 by the secure microprocessor 20 and the reverse. There may also be a requirement to transfer data from one location to another within the dual port memory 19. The address space occupied by the dual port memory may be any practical amount. Reset of the system microprocessor 1 initialises normal address decoding, with the dual port memory 19 inaccessible by the system microprocessor 1.

The execution of a TAA instruction, with for example X as the opcode, and the combination referenced as TAAX, is carried out if the system microprocessor 1 wants to move information from UCDPS memory to dual port memory 19, in which case buffers 2, 3, 4 would be activated by 9 for reads from any address in the UCDPS memory. During a write operation the address decoder enable signal 11 is active, enabling the address decoder 10 to decode a predetermined address block (that may be made programmable) of dual port memory 19 using chip select 13, that also keeps the buffers 2, 3, 4 disabled by blocking any enabling effect of 9 via logic gate 14. Data is read from UCDPS memory space and written to dual port memory 19. Instruction TAAY performs the reverse by activating 11 during read operations. Instruction TAAZ activates 11 for reading and writing. TAAB disables 11 for all reading and writing, the normal situation. The TAA instruction only affects operations that are fetching data, not instructions, and most system microprocessors have a signal to distinguish between the two. An instruction referenced as the TBAX instruction may be used to activate instruction fetches from dual port memory 19, by activating 11 during instruction fetches and may be disabled by the TBAY instruction. Instructions are read operations. TAA and TBA instructions may be used in any combination. A reset has the same effect as TAAB &

TBAY, ensuring normal processing on startup. While TBAX is active, instruction fetches from addresses outside the dual port memory 19 are from UCDPS memory. A watchdog counter or timer may be set, and this may be automatic to perform an automatic TBAY instruction or any other method to avoid trapping the system microprocessor in dual port memory 19.

This method and apparatus provides a novel transparent method of including one or multiple devices within a system microprocessor without potentially conflicting with existing resources in a UCDPS and has multiple applications to the art of system microprocessor design. To avoid problems with interrupts directing processing to another routine that expects a normal environment, interrupts are inhibited by TAA and TBA instruction. An alternative allows for similar instructions that do not inhibit interrupts, allowing the interrupt handler and or task switcher to handle the situation, in which case the TAA and TAB instructions are disabled by an interrupt and a record of their status is stored in a location, eg. a special register, accessible by the system operating system.

Secure processing is provided by including a second microprocessor 20 within 130 that may read and write to addresses within the secure address map 50 without being available to external analysis. Secure address block 50 is predominantly memory, divided into a small amount of mask ROM 51 to initially program the other information into the device, flash memory 52 for storage of information that needs to remain in the device in the event of total power loss, and battery backed static memory 53, that stores important information which may be rapidly erased in the event of tampering. The microprocessor 20 communicates with the secure memory 50 via address lines 84, data lines 100, and other various control lines including read write 93. Also decoded within the secure memory address is a battery backed realtime clock and or calendar 89 that cannot be tampered with and a crystal. A data encryption standard engine is preferably included. Decoding of secure addresses is provided by decode logic 25 and the various chip select signal are output on 83 to the various secure devices. The power management logic 65 receives external power on 60 and battery power on 87 from (preferably rechargeable) battery 70. An A/D converter 75 monitors voltage. Continuous power is supplied to 50 via 87. Power management 65 may also be used for any additional voltages to flash memory 52, other battery backed logic and provides recharging power to the internal battery 70. The microprocesor 20 communicates with the system microprocessor 1 via a dual port memory 19. The microprocessor 20 side 91 of dual port memory 19 is decoded by 25 via 40. Data lines 22, address lines 21 and read write 23 connect with 19 to allow reads and writes of information between microprocessor 20 and dual port memory 19. A similar method allows the system microprocessor to communicate with dual port memory via chip select 13 from its decode logic 10 and address lines 14 and data 6. The decode circuit 10 uses high order address lines 12 and control lines 32 (e.g.valid address) and 11 (activated by TAA, TBA). This provides a method of transferring information to and from external locations to dual port memory 19 that may be read and written by microprocessor 20. No user supplied program can access the information in secure memory without access to the secret codes required, and these may be made as complex as secure memory resources allow.

It is preferable that the secure microprocessor includes a direct memory access (DMA) facility to move blocks of information from UCDPS memory directly into secure memory locations and or from secure memory to external locations. This may actually improve the efficiency of the original system microprocessor, permitting it to perform other tasks while a block of information is securely processed in internal memory. Access to this DMA facility

should be decoded into the secure function address block and should only be able to be selected by an instruction originating within secure system functions (as described later). Any possibility of an external program and or a program executing in a user partition having unsupervised access to the DMA controller 125 that may be programmed to move a large block of system information to external locations would be disasterous.

The microprocessor 20 would usually program the DMA controller 125 via data bus 100 and chip select 142 and read/write 102, using a routine known to have originated within one or multiple predetermined system functions. The details of including a DMA controller 125 are not described or shown. The method involves multiplexing the address 5, data 6 and control lines 7 of the system microprocessor 1, with similar signals generated by the DMA controller 125 to read or write external locations and multiplexing of the address, data, and control lines of microprocessor 20 to read and write secure addresses. These methods are known to the art and, because the DMA controller is within the system microprocessor chip, arbitration logic between system microprocessor 1 and DMA controller 125 would be easier to implement at a logical level than for external DMA controllers. This type of DMA is transparent to external devices.

The invention also allows that the microprocessor 20 may be a duplicate of the system microprocessor 1 providing a very powerful processing system, allowing secure and unsecure execution to proceed concurrently. Another attractive option is to use two different system microprocessors e.g. an Intel type of CPU and a Motorola type of CPU. These may be multiplexed by one experienced in the art such that one system microprocessor performs normal system functions while the other provides secret processing of various functions. An electronic switch, that may be activated in any way, eg. hold reset low, may switch the roles. The secure functions may be duplicated, in part or whole, or each may have its own secure functions that are inactivated when a system microprocessor becomes the unsecure processor. A switch from secure processing to unsecure processing preferably ensures that any potentially secret information is flushed from CPU registers and any other locations that may become accessible to external analysis in the unsecure state. All secure functions would usually be inaccessible to the system microprocessor in unsecure mode. A person knowledgable in the art should be able to design such an embodiment that performs to the requirements of the invention. This provides a convenient means of providing an existing UCDPS with a means of integrating two different UCDPSs into one. Of course this scenario might be expanded to any number of system microprocessors within the one package. When multiple system microprocessors are included in the one package, the one that is normally associated with the resident operating system and peripheral arrangement in the UCDPS is referenced in this application as the Host CPU. Any other system microprocessors are referenced as a Grafted CPU. No changes would usually be required to any software to operate the Host CPU, however, other support may be required to simulate the correct environment for a Grafted CPU and one solution may be to include a programmable address trap for the grafted system microprocessor that detects all accesses to resources that need emulation.

It will be appreciated by those experienced in the art that the embodiment described with reference to Figure 1 may be readily transferred to a location external to the system microprocessor by providing a secure package and replacing the transparent address space of the version within the PCPU with an appropriate address within the UCDPS address space.

A basic embodiment of an SPD for use external to the system microprocessor is described with reference to Figure 2 of the drawings showing a printed circuit board 700 that is capable of connecting with an appropriate socket on the bus expansion of a UCDPS 720 via the gold fingers 701 on the printed circuit board 700. Mounted onto PCB 700 are an address decoder 702 to receive address signals from the address bus of the UCDPS 721 and various control lines 722 that it uses to decode the UCDPS side of the dual port memory 704 to a suitable address location in the address map of the UCDPS using chip select line 712. The lower order address lines 723 of the UCDPS together with UCDPS data bus signals 724 and a read/write signal 725 pass from the UCDPS bus via buffer 703 to the UCDPS side of the dual port memory 704 via signal lines 713.The part of 703 that buffers the data lines is bidirectional. A microprocessor 707 includes two interrupt lines 730 and 731 and an external address bus 714 and a valid address signal 733 and a bidirectional data bus 715 and a read/write line 732 and internal programmable non-volatile memory 708 (e.g. flash memory) and a boot routine 735 to load information into non-volatile memory 708. A static RAM chip 709 is connected to microprocessor 707 low order address lines of address bus 714 and the data bus 715 and read/write line 732. Static RAM 709 is activated by chip select 740 that is created by the address decoder 705 decoding the high order address lines on address bus 714 in conjunction with valid address signal 733. When static RAM 709 is selected the microprocesor 707 may read and write date to and from 709. The microprocesor 707 side of the dual port memory 704 is attached directly to the 707 data bus 715 and read/write line 732 and low order address lines of address bus 714. The microprocessor 707 side of the dual port memory is activated for read and write operations by chip select 750 generated by address decoder 705, from high order address lines on the address bus 714 and the valid address signal 733. A rechargeable battery 710 is included providing backup power via 711 to the microprocessor 707 and the static memory 709. When the the board 700 is plugged into an active UCDPS, the battery 710 is recharged from the system power supply. Microswitch 712 connects to interrupt line 730 causing an interrupt when the tamperproof enclosure 716 is disrupted. The tamperproof housing 716 securely encloses 710, 707, 709, 705, 704, 712, and all signal lines that may provide useful information. Interrupt line 731 causes an interrupt to 707 when the address decoder 702 decodes any address within the dual port memory, indicating that the external system microprocessor is accessing the device and that action may be required by microprocessor 707. The microprocessor 707 is normally in low power sleep mode. If awakened by interrupt 730 it immediately sequentially erases the values stored within SRAM 709 using a routine preprogrammed into 707 prior to enclosure in 716. If awakened by 732 it continues processing as required. The SPD as described may be integrated into a single chip. A person experienced in the art would be able to adapt this design to attach the SPD to any suitable non-bus interface. A suitable location may be the parallel port on a shared basis with the printer; the known art for other types of software protection devices describes such a shared interface. The inclusion of a cryptoengine implemented in hardware would enhance decryption processes that are fundamental to the secure and versatile functions provided by an SPD.

Figure 3 shows a block diagram of the address map for secure functions within the system microprocessor package/die 130 of Figure 1. These secure functions may only be addressed by the secure microprocessor 20 and may not be accessed by external programs other than said external programs providing information that is usually subject to validity checks and decryption before acceptance by the secure microprocessor 20 for further processing. The address decoder 25 decodes a battery backed real time clock calendar 89 with chip select 140, DMA controller 125 with chip select 142, Data Encryption Standard Engine 135 with chip select 143, and if the DES engine is

constructed in part or whole from programmable logic devices (preferably SRAM, that may be battery backed if non-volatility is required) that are dynamically programmed as required, these devices are selected by select line 141, tamper detect 80 (preferably including a continually powered simple microcontroller to provide continuous security monitoring) selected by 144, A/D converter 75 by select line 145, power management 65 by select 146. The preceding devices would usually have fixed locations in the memory space. They are part of the system functions and the chip selects 140,141,142,143,144,145,146, and any other additional select lines that may be included to access other secure devices, may only be selected if the instruction that outputs an address that is decoded to the preceding chip selects originates from within a memory location in the secure system memory 147, protecting the security of this area from non-system (user) programs - usually user application programs.. One method to do this is to latch the first address of an instruction and compare it with an address block that defines the boundaries of the secure system memory 147. This address block is preferably programmable to allow the size of secure system memory to be varied, however, there will be a known default on reset of the secure microprocessor 20. As an added precaution it is preferable to latch the first address of the preceding instruction and do a similar comparison. This requires any instruction that attempts access to secure functions in this part of the address map to have originated in secure system memory and the instruction prior to it must also have originated in secure system memory. This is to prevent a program that may be executing within a secure user partition from accidentally or deliberately loading the program counter of the secure microprocessor 20 with a value pointing to a secure function with unpredictable results. The address of the first instruction may be determined by including in the microcode of secure microprocessor 20 the generation of a signal to indicate that it is the first address of the instruction (this may already be the case). The program counter contents may also be latched. Chip select 147 from decoder 25 delineates the block of memory allocated to secure system functions. When the secure microprocessor 20 is reset it jumps to an initialisation routine in this memory. The size of this memory is preferably variable to accommodate changing circumstances. This is usually done by programmable boundary registers 160, that are selected by chip select 161. One boundary is usually fixed at the top of the available address space. The programmed value of 160 is supplied to address decode 25 and provided to its address comparators. These methods are well known to the art. Chip select 161 preferably requires the same precautions as regards checking the origin of the instruction as described for 140, 142, etc. Chip select 147 decodes the secure system memory. This preferably has the same requirements for two sequential instructions to have originated in secure system memory addresses in order to be decoded. An exception is reset or an interrupt that reset the latches that store the addresses of the two relevant instruction addresses to values that are within the secure system memory. This enables the secure microprocessor 20 to read information from its interrupt handlers. This also provides a method for a user routine to transfer processing back to system memory in a controlled way. A user function may write to an addressable location that generates a user interrupt 180; the system functions may then interact in any predetermined manner to meet the requirements of the user function. The balance of the secure memory is allocated to various user functions. In a multitasking UCDPS, this is preferably partitioned into multiple user partitions. The preferred method is to have one or multiple sets of address boundary registers 170, that may only be programmed by secure system functions decoding select 171, with the value programmed into 170 feeding back to the decode logic 25 to define the current user partition, that is decoded with chip select 148. This permits the available user partitions to be divided on a totally flexible basis as required. When processing transfers from one user partition to another, the secure system functions reprogram the appropriate values. When processing is transferred to a user partition no addresses are decoded outside this partition to prevent a user function

compromising the system partition or another user partition. If the program counter is loaded with a value pointing to an address outside the user partition, it will not be decoded and the user function will usually crash. In case of a crash within one of the user partitions a watchdog timer 190 may interrupt 191 the secure microprocessor 20 after a predetermined period. This is preferably a programmable period that may also be used to task switch secure processes in a multitasking environment. Prior to transferring processing to the user partition, the secure microprocessor 20 registers are preferably stacked and cleared of sensitive information and or the registers are duplicated. The dual port memory is decoded by chip select 150. The secure microprocessor 20 may also generate at least one interrupt 195 to the system microprocessor that directs the system microprocessor to an interrupt routine in dual port memory and or any suitable location. This location is preferably read only to the system microprocessor and may be read and written by the secure microprocessor 20. This interrupt may bypass any normal interrupts generated by the UCDPS to the system microprocessor and be processed transparently to the operating system. See known art US Patent 5274834. It may be used for any reason in particular to direct the system microprocessor to perform various functions within the UCDPS transparently to the UCDPS operating system. An interrupt may also be generated by the system microprocessor to the secure microprocessor 20. Interrupts to the secure microprocessor 20 are preferably specific to a particular source with sufficient interrupt lines to handle all interrupting devices.

Within the secure system memory is an area of masked ROM 51 that is usually a fixed amount, usually a fixed amount of flash memory 52 for storing information that survives total loss of power, and usually a variable amount of battery backed static memory 53 that securely stores secret system programs and data. This information may be lost in part or whole, due to accidental reasons, e.g. a flat battery (preferably rechargeable), or by activation of one or multiple tamper detect systems and or failure to comply with the conditions attached to using the SPD and or any other reason. System memory and user memory 54 is described later. Part at least of 53 and or 54 may be replaced by dynamic memory to provide greater memory density. This may particularly apply to secure system functions loaded from external sources as required, and user functions loaded as part of a PSO executing and or any other external information transferred as required.

Secure System Functions:

The system memory of an SPD must be preprogrammed with certain key programs and data prior to shipping to a user (usually as part of a UCDPS). This should be done in a secure environment, using secure methods, and is preferably completed during the manufacturing process. The service provider keeps a record of part at least of the information within each SPD. Once this key information is programmed into the system memory, any other types of programs and or data may be suitably encrypted by the service provider and transferred to a user's SPD (usually while within their UCDPS) using methods that maintain the security of the information. The suitably encrypted information is programmed into the system and or user memory of the SPD on a temporary or permanent basis, and in many cases this will be a transparent, dynamic process that occurs during the execution of various computer programs, particularly PSOs. This method allows almost any type of additional functions to be securely loaded and stored within the system memory, and or allows various programs to be loaded to update and or modify existing system functions and or any other transfer of information for any reason.

Secure system functions are those functions applicable to the correct operation of the SPD and the provision of required resources to multiple secure user functions. Secure user functions are those applicable to one or multiple PSO loaded into memory of the UCDPS that requires the SPD and system functions within the SPD for its correct operation. Secure user functions are usually an integral part of, or integrally linked with, a particular PSO and loaded into the SPD as required. A PSO that is supplied by the service provider to securely update secure system functions would usually act as a secure user function, although its effect is directed at secure system functions.

The preferred SPD consists of the following:

1. It provides a tamperproof environment which is not practical for an unauthorised party to penetrate for any reason including attempts at analysing or tampering with one or multiple secret processes that may be occurring within said tamperproof environment. This tamperproof environment may use a combination of secure packaging, using any known art to monitor the maintenance of the integrity of said secure packaging, together with a method of rapidly invalidating the contents should interference with the package be detected. As the preferred embodiment of the invention stores secret information independently of whether or not the UCDPS is active, part or all of the tamper detect and data invalidating methods preferably remain active on a continual basis. The preferred method is to have the secure microprocessor 20 (Fig 1) and or a microprocessor integrated into tamper detect 80 (Fig 1), continually powered and periodically awakened from a low power sleep mode to perform one or multiple housekeeping functions, including monitoring and or activating various intruder detect processes.

Secret information that may compromise the secure nature of multiple other SPDs is preferably stored in battery backed Static RAM (SRAM), a storage medium that may be rapidly invalidated by removal of power and or by a specially created subroutine that cycles through the memory changing values and or a specially designed cascade system that triggers automatic invalidations of static memory storage elements as is known to the art (reference Dallas Semiconductors Secure Microcontrollers). The invention allows for any known method and apparatus of detecting physical tampering with the SPD and allows for any method and apparatus of invalidating secret information in any type of memory storage device.

Secret infomation that is only likely to compromise the security of a particular SPD may be stored in SRAM, however, information that should survive invalidation of the information within SRAM is preferably stored in non-volatile locations. When this information needs to be programmed and or reprogrammed dynamically in the normal course of operation of the SPD, it is preferable to use flash memory or an equivalent. When the information does not require alteration after initial programming it may use any type of non-volatile memory storage device.

Information not requiring secrecy (as far as practical) and that is consistent across multiple SPDs is preferably implemented in mask ROM during the manufacture of the SPD. This usually includes initialisation routines to program other information into the SPD. When constructing an SPD that is not within the system CPU, the CPU chosen for the SPD will usually already have a boot or initialisation routine embedded within. Those experienced in the art will appreciate that information stored as masked ROM inside an integrated circuit (IC) package may be analysed, however, this is usually with great difficulty.

Where certain unique feanures are required in each SPD at the time of manufacture and secrecy (as far as practical) is not essential, they are preferably implemented by laser programming of masked elements. This usually applies to one or multiple passwords that are applicable to a particular SPD.

The secret processing device (SPD) is a device that is not practical to tamper with. This device contains various secure functions that may perform useful functions for suitably configured software objects. It also provides various secure functions that permit a provider of protected software objects, referred to as service provider, to create an effective method of renting software to users. A number of alternative methods of securely distributing software are discussed. The method is secure from the perspective of the producer of the software object and provides a convenient means for a potential user to have access to a large amount of software that they only pay for as they use.

The invention allows that attempts may be made to physically tamper with the SPD. This may be for any reason, including the unauthorised extraction of secure information from the SPD. Secure system tamper detect functions, using any method and apparatus, may be used to detect tampering and or to take direct (that preferably includes immediately erasing and or altering information within part or all secure storage devices) and or indirect (e.g. via error functions) action in the event of tampering. Part of the tamper detect functions allow for any method and apparatus, referenced as secure system continuity functions to confirm that one or multiple of any tamperproof mechanisms remain intact. One method is to include bidirectional logic at each end (or any other location) of the various signal lines to check for continuity of signal traces and or functioning of attached logic elements in those instances where the normal function does not permit this. This bidirectional logic is usually connected, directly and or indirectly, to addressable elements under the control of suitable software routines. The invention also allows for any method and apparatus to detect loss of clock to the realtime clock/calendar and or any one or multiple other clocked elements, including routines that periodically read these clocked devices (directly and or indirectly) to ensure that there are the expected incremental changes secondary to an active clock. It is preferable that part or all of the tamper detect mechanisms remain functional when the system power supply is removed. This may include using battery power to maintain one or multiple microprocessors within the device in an operational mode, enabling them to execute various system functions. Loss of battery voltage below a predetermined threshold (as detected by an integrated A/D converter) may trigger the erasure of part or all secure elements. It is preferable that an independently timed function is implemented (e.g. RC network) that must be periodically refreshed by one or multiple microprocessors. This confirms the presence of an active CPU and failure to periodically refresh this function would usually cause a default erasure and or alteration of secure storage elements.

The invention allows that various errors and or validity failures and or any processing error and or any other event may be recorded by secure system error monitoring routines (usually implemented within secure system memory). These may perform any functions, that may include:

recording abnormal events; and or

in response to a predetermined number and or types of abnormal events (and or any other reason) take one or multiple actions (that may be any action, including calling other functions to partially or totally disable the device); and or

return processing to the system CPU (with or without error reporting).

There may be a requirement to disable part or all of the SPD and or part or all of other apparatus that the SPD may be integrated within (e.g. system CPU). The functions to perform this are referenced as secure system disable functions and they may be implemented using any method and apparatus, including:
the generation of various clocks (and or any other meaningful signals) that trigger immediate erasure of volatile elements; and or
setting/clearing of flags (preferably in non-volatile locations) that may be read by various other functions that will not continue (and or any other outcome) in the event of an unacceptable value within a flag.

The invention also allows for any method and apparatus that may temporarily prevent, in part or whole, action by the disable functions. This may be for any reason, however, the primary one is to stop inadvertent triggering of these functions during software development. The invention allows for any method and apparatus that prevents infringement of system security when the disable functions are in part or whole temporarily inactive.

2. It provides one or multiple blocks of memory arranged in a manner that prevents unauthorised analysis of the contents of such memory unless intended. This memory is referred to as secure memory. This may apply even if part or all of the memory contains information that is not secret.

The memory blocks may use any types of memory storage device, in any mix and combination. There are preferred types of memory storage devices to meet the requirements of specific functions.

The primary purpose of secure memory is to provide part of an apparatus that, when combined with a secure method of processing information within the secure memory and a means of transferring information between the SPD and external locations, allows certain secret processes to occur and or certain secret information to be securely stored. The processing of information within secure memory may include the use of any mix of secure and unsecure programs and or data, and any interaction with resources that are external to the SPD.

An SPD usually has one or multiple blocks of memory storage devices that may consist of any type and combination of memory storage devices arranged to make it not practical for unauthorised parties to analyse the values stored within part or all of said memory storage devices.

The memory storage devices preferably:

(a) include one or multiple blocks of Static RAM that are made non-volatile by connection to a non-disruptable power source that is preferably a rechargeable battery integrated into the device and or its enclosure, and or a rechargeable battery external to said device, and said Static RAM is used in part or whole to store secret information that should usually be invalidated in the event of any tampering with said device, and said Static RAM is preferably connected directly and or indirectly with one or multiple methods and apparatus to detect said tampering and invalidate and or activate invalidation, of part or all of said secret information as a result of said tampering. The

invention also allows for the inclusion of any method and apparatus to invalidate in part or all secret information stored within said static RAM for any other reason. This memory usually stores:

(i) secret system functions implemented at least in part as software routines, that need to be maintained in secrecy (as far as practical) and that cannot be stored in encrypted format in an external location and loaded and decrypted as required. An example of this may be the master decryption algorithm and or keys. If this was loaded from an external location it may be analysed and used to break the security of other encrypted information. Partial loading of decryption algorithms may be possible as long as sufficent function is kept securely within the SPD. Said sufficient function may in part or whole be a hardware implementation of a decryption algorithm.

(ii) information that may or may not need to be secret that is required to correctly interface with externally available information, this may include the loading of other information.

(iii) information that it is determined, for any reason should be within the SPD on a continual basis.

(iv) information that is loaded from external resources. This may include additional secure system functions loaded in encrypted format and subsequently decrypted and may include appropriately encrypted objects supplied by an authorised party to modify information within the SPD.

The information described in (i), (ii), (iii) and (iv) constitutes part of the secure system functions (53 of figure 3) and consists of information that is known to be available within, or able to be loaded within, the device when required to perform the functions that are an integral part of the SPD. System functions are also known to have been carefully prepared and scrutinised in a secure environment to ensure that they do not corrupt and or compromise the secrecy of information within the SPD. Those secure system functions that are loaded into the SPD in encrypted format usually have tamperproof validity checking processes integrated into their structure to ensure the validity of the information prior to associating it with other secure system functions. That part of the secure memory that includes secure system functions is referenced as secure system memory.

(v) other information that may be loaded into the battery backed SRAM and may include one or multiple secure user functions (54 of figure 3). These are usually software objects supplied by various producers that have a requirement for interaction with the SPD. They usually require appropriate conversion of the software object by an authorised service provider to one that may be recognised and processed by the SPD and such an object is usually referenced as protected software object or PSO. A PSO is usually encrypted and preferably has appropriate validity checking mechanisms included to ensure that the information is as supplied by the service provider. Those parts of the PSO that are to be transferred to locations within the SPD, whether data and or computer instructions, are referenced as secure user functions. In applications where this information is data that is to be processed securely using secure system functions, accidental and or deliberate tampering with the data usually has no potential unwelcome consequences within the SPD as the processing is performed by known processes.

(b) static RAM (SRAM) that is not battery backed and or dynamic memory may be used for secure system functions described in the preceding (a) part (iv), and or secure user functions in (a) part (v), and or any other information loaded into the SPD.

(c) an area of programmable and or reprogrammable memory that remains non-volatile when all power is lost. This preferably includes one or multiple blocks of intrinsically non-volatile and reprogrammable memory e.g. flash memory and or EEROM, including any required componentry to support programming, erasure and reprogramming of said flash memory and or EEROM. Particular applications of this area are the storage of information that should survive an erasure of SRAM for any reason, including accidental erasure. One of the features of the SPD is its capability, with appropriate software, to select random encryption keys and validity check sums, and use these to encrypt information stored externally, preferably on a mass storage device. This information may need to remain retrievable if the SRAM contents are corrupted. By retaining the keys to this information in non-volatile locations, a suitably protected routine may be used to retrieve this information by the service provider. It also prevents tampering with externally encrypted information as the decryption key is inaccessible and may be varied every time.

(d) includes one or multiple blocks of memory of mask ROM that is programmed at the time of fabricating the memory storage devices and said mask ROM preferably includes an area that may be customised to create unique information for each device, one method of customising the device is with a laser. This is usually used to initially program data into other storage devices.

The current system functions within an SPD preferably have a version number stored in an externally accessible location, eg. dual port memory 19 of figure 1 that may be read by PSOs to ensure the SPD has the necessary resources to meet the requirements of the PSO.

3. It provides at least one secure microprocessor 20 and a method of decoding part or all of the secure memory and any other addressable functions (e.g. timer, realtime clock, decryption/encryption engines, interfaces, etc) into the address space of the secure microprocessor 20. The microprocessor is designed such that secret information that it reads and or writes and or processes, in part or whole, is not exposed to unauthorised analysis.

The secure microprocessor 20 may be continually powered to perform reliable tamper detection and invalidation. The power source is usually shared with the battery backed SRAM and where present, the realtime clock calendar.

It is preferable that the reset line on the secure microprocessor is connected to the reset line of the host UCDPS, enabling it to perform error checking on internal stored information prior to performing functions required by the UCDPS.

The secure microprocessor on reset (and or any other appropriate event) and or as part of its normal functions may perform various housekeeping duties while waiting for one or multiple intrerrupts generated by the UCDPS, and or the reading of one or multiple appropriate values from one or more polled addresses, that may also be directly and or indirectly written to by the system microprocessor, and or any other method that activates the microprocessor and or any one or multiple other functions of the SPD to further interact with the UCDPS as required.

4. The SPD predominantly is a secret processor of information and a secure and secret repository of information, that in part or whole is generated (including by decryption) within the SPD. It is an essential function that there is a

means of transferring information in and out of the SPD without compromising the security of information that must remain secret. This entails two basic requirements:

(a) The provision of one or multiple physical interfaces between SPD and sources of information. The invention allows for any known interface. This includes information that is transferred via the bus of the UCDPS, that is the usual method when the software objects using the SPD are executing and or being processed by the system microprocessor, and or information entering through one or multiple ports that may be read by the secure microprocessor and or any other function within the SPD.

The preferred interfaces include any ports that are part of the secure microprocessor or any other part of the SPD, dual port memory 19, latches and or registers (unidirectional and or bidirectional), FIFO memory, a facility for the secure microprocessor to have direct access to the address bus of the UCDPS and move information under programmed control and or by direct memory access (DMA).

(b) a method for the SPD and UCDPS to determine which locations have valid information and a method of acting on this information. The information may be commands and or programs requiring execution and or data for any reason and or any other information. This is a function of the secure system functions and specifically those referenced as secure system I/O functions. They require similar processes to those provided by any operating system and are within the expertise of those experienced in the art of writing operating systems. Moreover, as the SPD includes functions to load and execute externally supplied software objects that may securely modify the various secure system functions, more flexibility is provided with an SPD than many UCDPSs having part of their operating system in memory that is not easily modified.

The preferred embodiments of the invention provide a dual port memory 19 that is accessible by the secure microprocessor and the system microprocessor. This occupies a predetermined part of the address map (that may be programmable) as previously described with reference to Figures 1 and 3.

The next part of the description may be better understood by reference to Figure 4 of the drawings that shows:

A system port structure 199 is established that may have one or multiple addresses which the system microprocessor writes to, referenced as system command input port 200 and one or multiple addresses that it reads from, referenced as system command output port 201. The SPD reads command input ports and writes to command output ports. As these are usually part of a block of memory, they may be dynamically reconfigured by appropriate interaction between system microprocessor 1 and secure microprocessor 20. This reconfiguring may change locations and or the number of addresses constituting a port. It is preferable to have a system input data port 202 for the transfer of information other than commands from UCDPS to SPD and a system output port 203 for non-command transfers from SPD to UCDPS. In the case of dual port memory a large block of addresses may be allocated for non-command information and the addresses and sizes may be dynamically configured. The actual allocation of input and output ports is preferably a function of the SPD and is likely to be a dynamic state. In a single tasking environment this may be the only interfacing required. The inclusion of a DMA channel 125 on the SPD is the preferred method of moving

large blocks of information in and out of the secure memory 53, 54 of the SPD. Address and control lines 220 and data lines 221 from the DMA controller 125 are multiplexed with similar signals from system microprocessor 230 are multiplexed in 235 for interface with external memory. Address and control lines 222 and data llines 223 are multiplexed (not shown) with similar signals from secure microprocessor 20 for transferring information to and from secure memory 53 and 54.

The invention also allows for the SPD to handle the requirements of multiple PSOs in a multitasking environment and that the system command and data ports as described may be sufficient if the UCDPS operating system is modified to send a command to an appropriate location in a command port to instruct the SPD of a task change and does not proceed until the command is acknowledged.

The preferred method is to use the system command and data ports for establishing certain parameters within the SPD when a PSO first requires access to the SPD. The PSO would usually send information requesting a user partition 54 of Figure 3 and a user port structure 205 of Figure 4. The SPD would usually respond with availability of this memory and dynamically configure a user command input port 206 and or user command output port 207 and or user input data port 208 and or user data output port 209. The PSO stores these port addresses in a suitable location in its own address space and directs all commands and other information to and from these user ports until otherwise appropriate. A multitasking kernel within secure system functions is preferably responsible for such port configuration as part of its functions. Additional PSOs create there own user ports, e.g. 210 and 215 of Figure 4. The space used by these ports is reallocated when a software object terminates interaction with the SPD. Any one or multiple user ports may be dynamically reconfigured as required while still in use with a particular PSO. This process permits the SPD to be transparent to the UCDPS task handler.

5. Secure System and Secure User Partitions:

If the SPD is to provide any useful processing of information supplied, it requires a method of transferring information into secure areas where it may be further processed. As described, a potential unsecure process is introduced into an SPD once the facility is provided to load externally supplied information into secure memory that in part or whole consists of executable code. PSOs that are to modify the secure system functions are usually provided by the service provider from software objects in their control and the security is good. When a PSO is produced by a Producer, there can be no such guarantee of the integrity of the contained program code. The execution of this material may read information from secure system functions and write it to external locations. In a multiuser system, it may also compromise information relevant to another PSO.

The preferred method is to partition the available secure memory into paritition as previously described that includes a system partition and one or multiple user partitions. Programs within a system partition may access any secure memeory address. Programs within a user partition are confined to their own partition. This is implemented using dual latching of instruction sources as previously described. This protects system integrity and the integrity of one user partition from any other. An alternative is to perform this function with software, by checking that each instruction executing within a particular user partition is not intended to make an unauthorised access to system

memory and or other user memory. Another solution would be to allocate a separate microprocessor to one or multiple user functions.

When the secure system kernel switches processing between user functions, it programs logic with the address boundaries of the current user partition that is compared with an instruction. A separate user partition is allocated to each user function.

The invention allows for any method and apparatus that prevents any particular user function from accessing, in an unauthorised manner, secure information within system partitions and or other secure user partitions. The method does allow valid transfers of processing across system and user functions.
It is preferable that the size of the partitions may be varied, preferably under the control of secure system functions.

6. Initial Programming, Reprogramming and Erasure of secret information:
The invention allows for secure system initialisation functions (SSIF) that may use any method and apparatus to initially program secure system functions into secure locations within the SPD, preferably into battery backed static RAM. This usually occurs prior to release of the SPD from a secure environment. The SSIF are part of the secure system functions, however, they include information that it preferably not made public, however, the invention is not compromised should this occur. For this reason they are suitable for use in mask ROM. Any other secure system functions may be included into mask ROM, however, this is not the preferred location for any information of a sensitive nature. It addition to security factors, the inclusion of the majority of secure system functions in reprogrammable storage elements allows them to be readily updated. The invention allows that that the SSIF may be used later to erase and or modify and or reprogram the SPD at a later date. The invention also allows that part or all of the functions within the SSIF may be called by other secure functions as part of the normal operation of the SPD. For example the routines to load information from external locations and to program information into flash memory have obvious multiple uses. Certain provisions within the SSIF should only be capable of use when it is known that secure information within the device is invalid.

The preferred method and apparatus is to store the Secure System Initilialisation Functions within (preferably secure) storage locations prior to encapsulation (that may be the package of an IC and or any other additional packaging) of the device at the time of manufacture. As a minimum, the SSIF information included within the device at the time of manufacture should be sufficient to load and or program other information into the device and where necessary initiate processing of said other information. This provides an SPD that may then modify itself as required. Said other information may be any information and may include additions to the SSIF not included at manufacture. The storage locations should retain SSIF functions (in part or whole) when other information within the device is erased for any reason. The SSIF may include any required support hardware to program particular storage devices, eg. charge pumps and or supply of special voltages and or timers and or glass windows to erase EPROM. The SSIF is usually implemented within secure memory (that is preferably mask ROM, however, it may be any suitable type of storage device) and usually includes functions:

to respond to a command to activate one or multiple SSIF functions (and or any other necessary commands); and or

to retrieve externally supplied information, that may use any method and apparatus provided for in a particular SPD and to program part or all of this information and or any other information into required locations; and or to finish programming; and or to verify that the programmed information is error free; and or to terminate the process such that various applicable functions remain available as secure system resources; and or to direct processing to part of the information that has been programmed (and or otherwise initiate access to this information).

(The ability to load information and subsequently direct processing to this information is a key aspect of the invention. With the addition of a suitable decryption method within the SPD, the SPD may load encrypted information, decrypt this information and then direct processing to said decrypted information. The addition of routines to pass information back to external locations completes the process. These functions are described in detail later).

The SSIF and any subsequent secure system functions may load information from any relevant external location to assist the process and or may call routines within external locations to assist the process.

Any SSIF function that allows programmed information to be read back for verification (and it is preferable that this occurs) may use any method and apparatus to prevent a user from activating this function at a later date and possibly being able to access secret information. The preferred method flags a non-volatile programable location once the readback process is complete in a manner that does not leave said flag clear in the event of a partial readback. The preferred method to prevent the flag remaining clear in the event of a partial readback is to activate a watchdog timer that times out after a predetermined interval and sets the flag preventing further verification readback by trigggering a flip flop. It is preferable said flag can only be cleared after secure storage elements have been erased and or otherwise suitably modified. This is not a function that should be available in unsecure environments.

Disclosure of the information constituting the actual SSIF is unlikely to jeopardise the security of other secret information, however, it is preferable that unauthorised parties are prevented from initialising and or erasing and or reprogramming the device and any method and apparatus may be used to implement this. It is preferable that these processes are password protected (using any password system) against unauthorised use.

One method of implementing SSIF would be to serially clock the required information into the device via latches (that may require a certain predetermined sequence to activate the process). This may not require any predetermined software routines within the device.

The preferred method uses a secure software routine executing from within secure ROM that uses the Timed Password Access process described below to activate programs that perform the functions previously described for a SSIF, transferring the relevant externally supplied (and usually secret) information to the relevant internal storage devices and subsequently initiating processing of this information.

The actual method of programming information into the storage devices will depend on the type of storage device and may use any known method.

The timed password access method makes it unlikely that the password protection will be defeated, while retaining functionality for those parties with the necessary knowledge, even in the presence of previous unsuccessful attempts at programming and or deliberate attempts to inactivate the device (eg. computer viruses). This contrasts with password systems that permanently inactivate the process after a predetermined number of attempts, possibly preventing further programming of the device by authorised parties.

The invention allows that a preferably unique password is programmed (usually as part of SSIF) into each device. Without access to this unique password the probability of unauthorised activation of SSIF is not a practical outcome.

In an SPD integrated within a system microprocessor, particularly one with multiple microprocessors within, the SSIF may reside in memory locations exclusive to one of the on chip CPUs and be transferred where necessary, using any internal mechanisms (including software), to any required storage devices; and or
may be loaded into memory locations shared by multiple CPU's within the package;

and or may be loaded into multiple locations, each location of which is exclusive to a particular CPU within the device.

The invention allows that only one CPU or a subset of available CPU's may load information for other CPU's, and or that particular CPU's load information for their own use.

The preferred method of activating the SSIF functions when the SPD is within the system microprocessor is to load the password into one or multiple CPU registers and execute a specially created instruction that that activates SSIF to read the password and continue as appropriate. An alternative is to include the functions that detect and process the post instruction symbol stream as described later.

The timed password access (also referenced as TPA) may use any method and apparatus. It prevents any practical gain from attempting unauthorised access to any particular password protected event. It is based on a password of such complexity that in practice it would take such a long time to try all the permutations that it is not practical to gain access to the protected event. Said complexity is assisted by incorporating a delay mechanism that restricts the frequency of attempted access. Said delay may be variable for any reason (e.g. to allow for legitimate errors) and may be created using any method including software loops and or physical delays. The delay may be a hierachical system that includes different delays depending on the number of incorrect attempts at access. It is preferable that said delay is unaffected by powering down of the device to prevent rapid power cycling defeating delay mechanisms. One method and apparatus consists of the following steps:

a) create one or more password keys that are stored securely.

b) create a means to store a cumulative count in a device that is reprogrammable and preferably non-volatile.

c) create a means to generate a known time interval. The invention allows for embodiments allowing a variable interval, this is most readily achieved by a software loop.

d) create a means to input a password, eg create a specific instruction that can pass externally supplied information to the relevant routines.

e) create a means to input function required should password succeed (not required if only one option).

f) user activates d) and e) including transferring password and target function to the process.

g) check the value in cumulative count in b).

h) if less than certain predetermined value then go to step j) else proceed.

i) invoke c) to generate time delay.

j) increment the value in b).

k) confirm step j) has occurred if there is a chance that external influences may interfere with j).

l) input password using d) and compare with key in a). If a match go to step o), else proceed.

m) set flag in external memory to indicate failed attempt at calling program.

n) exit, to try again enter at f). (if predetermined count above c) retry will be immediate, otherwise a delay will be encountered every time).

o) clear flag in external memory to indicate success.

p) proceed with called process.

q) return to external memory when finished.

Note: for passwords that protect access to processes that are implemented after destruction or alteration of erasable areas, software routines and associated key codes should be stored within memory that is not erased.

The advantage of TPA over a limited number of attempts that then blocks the system, is that it prevents the accidental and or deliberate permanent disablement of part or all of the device. The invention allows for a mix of methods.

Electronic Signature: One or more processes during manufacture and or initial programming and or normal operation of the invention may need to identify parameters unique to a particular PCPU and or ESPD and or unique to a particular group of PCPUs and or ESPDs (for any reason, including for example, referencing a secure database to determine a password to activate the initialisation program described above). This may be done by any method known to the art including physical markings on the outside of the CPU package, however, the invention allows for one or multiple serial numbers and or any other identifying symbols to be included within the device, usually at the time of manufacture. These are amenable to retrieval under program control and or any other form of automatic process using any method and apparatus. This provides an automatic method of uniquely identifying a particular device and or group of devices. This is referenced as an electronic signature and is usually included as part of the SSIF. Said one or multiple electronic signatures may be transferred to an external location using any method and apparatus and used by an authorised party as an index to secure information stored within that particular device (and or for any other reason). The preferred method when the device is a PCPU is to create a specific instruction that when executed stores said serial number from a non-volatile storage location within SSIF to a predetermined CPU register. This process is usually accessible to anyone, although it may be protected by passwords and or any other method. For ESPDs the serial number is usually read from an addressable location within the ESPD by the system CPU. In the case of the ESPD described with reference to figure one, the secure system interface functions

programmed into flash memory 708 would include the electronic signature and when the microprocesor 707 is first activated by an interrupt on 731 after programming of said secure system initialisation functions, a routine would transfer the electronic signature to a predetermined location in the dual port memory 704, where it is accessible to the system microprocessor.

The invention allows that a secure system user password function may be included within one or multiple PCPUs and or one or multiple ESPDs and this may be required to activate part and or all of the invention. In the case of a system CPU it may also be required to enable the normal processing functions of the device, providing a secure method of stopping unauthorised use of the UCDPS containing said system CPU. Any method and apparatus may be used to implement this function. The usual presence of programable memory and programable non-volatile storage elements provide for a plurality of methods. The invention allows for a multi-tiered password system. The preferred embodiment is a time based password system (as discussed elsewhere) that resides in secure system memory and activates routines that reverse various locks placed on part or all of the device.

The password functions usually include routines to disable part or all of the device in response to a specific command, a method that requires the user to specifically disable the SPD, and preferably requires entry of the correct password; and or functions (usually implemented in hardware) that disable part or all of the device in response to reset and or power down and or any other criteria including automatic timeout (preferably programable), the password processing system is not usually disabled; these functions automatically disable the SPD and or other applicable devices and require the correct password to reactivate the SPD and or other applicable devices.

The password(s) is usually stored in secure non-volatile system memory. The device may be shipped to the user with a known default password and or the password system disabled. Entry to the password system may use any method. In the case of a PCPU this may include use of a special instruction and or a suitable Post Instruction Symbol Stream (PISS). In the case of a ESPD it may involve passing commands using one or multiple methods as described elsewhere in this application, usually by writing and or reading predetermined address locations. A user accessing the device with the correct password may be able to change passwords.

The password system is usually constructed to allow the service provider to reinitiate or disable said password system by supplying an appropriate software object, preferably a PSO.

The inclusion of at least one unique and secure code within each device together with other suitable support resources allows a plurality of methods of secure information transfers to be established between an information provider with access to the secure contents of the device, and or provides for the secure transfer of information in the reverse direction, and or permits information to be specifically encrypted for a particular secure system. These are referenced as system local code functions and they assist the implementation of multiple secure applications, including the secure transfer of information to a device that can verify the source and or validity of the information, and or the secure supply of information from a particular device that the can be verified for validity and source by an information receiver (with access to the secure information within the originating secure system CPU); this may be used for any reason including secure communications and or the secure transfer of electronic funds.

The inclusion of one or multiple system group code functions that are identical across a particular group of devices (e.g. those destined for the same country) may be used for any reason. This may include the restriction of certain PSOs to particular group codes. One or multiple group codes may be common to all SPDs. The invention allows that part or all of group codes may be user programmable and or password protected. This may allow, for example, parents to restrict childrens access to particular PSOs.

The secure local and or group codes may be data and or actual computer instructions.

The effectiveness of the software distribution system forming part of this application is partly dependent on a service provider having access to secure information within each SPD and that some of this information is common to multiple SPDs enabling creation of PSOs that have general application, and that some information is specific to a particular SPD.

The inclusion of secure system command functions to detect instructions (that may be implied instructions) amongst information supplied to the SPD (using any method and apparatus) and or generated by a secure user function and or generated by secure system functions requesting the SPD to perform certain tasks. These tasks may be any and may include:
commence execution of internal programs from any source; and or
pass data received from external sources to internal functions; and or
receive a request from internal functions to transfer processing back to the system CPU for any reason; and or
accept data from internal functions for tranfer to a location readable by the system CPU; and or
provide a command structure within the SPD to co-ordinate other system functions and, where appropriate, interact with secure user functions; and or
where applicable, co-ordinate interaction with realtime decryption processes; and or
any other required function.

The invention allows for any method that permits an SPD to monitor a PSO as it is executed in order to detect various specially constructed process transfer instructions and or other suitable markers that indicate that interaction with the SPD is required. This particularly applies to a PCPU, where the method usually involves the transfer of processing from external unsecure memory to internal secure locations for continued processing by the system microprocessor using secure methods and or by other embedded microprocessors (that may include other system microprocessors, and or the activation of realtime decryption use encrypted information in external location.

The process transfer instruction may inherently direct external programs to the appropriate internal function or may require a post instruction symbol stream as described with reference to the preferred embodiment.

Secure system command functions also include any functions to transfer processing back to the appropriate PSO.

The secure system command function should be structured so that entry to secure system functions is in a regulated manner. This is readily achieved for an ESPD where interfacing may be directed to a limited number of addressable locations that may have various validity checking performed on the data. The process is more complex for a PCPU and described in more detail with reference to a PCPU.

An important function of secure system command functions is to direct the decryption of incoming encrypted information, direct the transfer of the decrypted information to a suitable location and where this decrypted information consists of computer instructions, direct execution to the relevant starting point in the decrypted program and provide any necessary support functions as said computer program is executed. When the incoming encrypted information is data this should be processed as required, which may include appropriately linking it with any internal and or external programs and or data and or special purpose functions (e.g. the data may be used to configure programable logic, creating its own decryption engine) including a linked computer program also transferred in encrypted format. The command functions also direct the return of execution and or data to external locations as required.

7. The invention also allows that one or multiple hardware devices within the SPD may actually be fabricated in part or whole from programmable logic devices. This particularly applies to encryption/decryption engines that may be dynamically engineered as required. The preferred type of programmable logic is that known to the art (refer to programmable gate arrays by Xylinix) using battery backed static memory to create the interconnections between various logic gates, as this may be rapidly erased if required. The information to transfer this information to the programmable logic elements is preferably via one or multiple addressable locations, and is preferably parallel data. Part or all of such devices may need programming prior to leaving a secure location.

8. Secure Decryption, Secure Processing, Secure Decryption and Processing, Secure Processing of Information Unique to the SPD. The system functions should provide suitable software routines such that, when requested by appropriate commands, they perform a combination of functions that affect any combination of the following:

- for the secure transfer of at least a portion of encrypted information constituting part or all of a software object from a location external to said physical device, to a location internal to said physical device, wherein said physical device securely decrypts part or all of said encrypted information within said physical device in conjunction and or subsequent to said transfer and
- may initiate and securely process part or all of the ensuing decrypted information in conjunction and or subsequent to the decryption process and
- may interact in any way with any other internal and or external information to correctly said process and may terminate said process as required and
- said terminate may transfer data and or execution to any other internal and or external location, including the external software object and
- the preceding processes occur in a manner that minimises or eliminates analysis of part or all of the decrypted instructions and or data; and or

- that includes computer instructions and or data securely programmed within said physical device and a facility for an external software object to transfer processing to said computer instructions and or data securely programmed within said physical device, and the capability of processing part or all said securely programmed within in a secure manner, interacting in any way with any other internal and or external information to correctly said process and
- may terminate said process as required and
- said terminate may transfer data and or execution to any other internal and or external location, including the external software object and
- the preceding processes occur in a manner that minimises or eliminates analysis of secret information; and or
- with the capability of being suitably requested by an external software object to provide information securely stored within.

The secure system decryption/encryption functions (together with the necessary command functions to load encrypted information and or to execute, and or otherwise manipulate, the information decoded from this encrypted information, possibly in conjunction with clear code and or other decoded information) may eliminate the requirement to preload specific secure user functions into the device prior to supplying said device to a user. Instead each PSO may include the secure user function as encrypted information included within the PSO supplied to a user, resulting in a device that can securely process part or all of a diversity of software objects. As suitable system command functions may be constructed to dynamically load blocks of encrypted information in and out of secure user (and or system) memory, much larger portions of encrypted information may be utilised as part of a software object than is the case with devices dependent on secure information preprogrammed into a limited amount of secure user (and or system) memory.

In addition to decrypting and executing the equivalent of secure executable user functions, the invention also allows that the device may securely add to and or edit secure system functions using a similar process.

The invention also allows for part of the secure system functions to be loaded (usually in encrypted format) into the device from external storage each time a UCDPS is booted (and or on any other basis).

The security of the secure system routines and in particular secure system decryption routines stored within the SPD is pivotal to maintaining the security of processes using the device. The information within secure system functions must be protected to a level that makes it not practical to defeat and while any storage device may be used to retain the secure system functions within the device, the preferred method uses battery backed static memory. This can be rapidly erased in the event of tampering. and such a requirement particularly applies to any system functions that are stored in decoded format.

The transfer of information from one location to another may result in transmission errors and the invention allows for secure system error detection functions that may use any known method and apparatus to detect and or correct these errors.

As the usual location of the SPD is within the UCDPS, information that is to be transferred to the SPD may be accessible and deliberately modified, e.g. computer viruses and or attempts to reverse engineer the SPD. The invention allows for secure system validity checking functions, that may use any method and apparatus to verify that the information supplied to the SPD is as intended by the information provider, and or take any required actions that may include directly or indirectly (usually via secure system error monitoring routines) disabling part or all of the SPD. Where applicable, this may include the erasure and or alteration of secure information.

The use of cyclic redundancy checking (or CRC) of information generated by a service provider and embedded within a PSO and then encrypted is one method of providing secure validity checking functions. The reversal of this process in the SPD may use any combination of hardware and software methods. The process is well known to the art.

9. Secure system decryption/encryption functions: The decryption functions may in part or whole be implemented in software to decrypt externally supplied and encrypted information using any known methods, including the data encryption standard. One or multiple hardware based encryption/decryption engines may perform the decryption, in part or whole. Such an engine is one compatible with the Data Encryption Standard (DES). The method of using predetermined processes located within the SPD to decrypt (and encrypt) information is referenced as the Standard Decryption Process in this application. Standard Decryption Processes may require the supply of various codes to function correctly. The original cryptography processes were developed for the secure communication of information between parties and they work well when this is the primary motive. When the purpose of encryption is to enable one party, in this case the producer, to encrypt information to protect it against unauthorised use, and the second party is a user who may prefer that the information was not encrypted, then the original basis for secure cryptography changes, and the premise for security is based on the fact that said second party will receive information, however it will be difficult for them to access it in clear code. This has resulted in various specialised devices to decrypt information. As described this method does not provide a system that is 'not practical' to defeat. The Oscar method of secretly decrypting and executing information provides a method that is not practical to defeat.

The capability of supplying an SPD with a PSO that can be made to perform any desired function within an SPD that is consistent with available resources and constraints of said SPD, allows said SPD to be dynamically modified to perform any function as required. This permits a PSO and or any other internal and or external function to actually request one or multiple decryption functions to be loaded into the SPD. Said decryption functions may include information that is used to dynamically manufacture a hardware decryption engine from programmable logic within said SPD.

The capability of significantly varying the decryption process, and or constructing hardware cipher engines from volatile electrical connections that cease to exist when subjected to analysis, and or dynamically engineering cipher engines to suit a PSO makes characterisation of the decryption process very difficult. The known art does not describe such a method and apparatus, which this invention references as Dynamic Decryption in this application.

By including one or multiple decryption processes within an actual PSO, the decryption process can become self modifying with the instructions of the actual PSO varying decryption parameters and or decryption algorithms and or installing, in part or whole, one or multiple new decryption algorithms during the process of executing the PSO that are further used to decrypt additional parts of the PSO. This may occur on multiple occasions, in any combination, during execution of the program. The key to this process is to include with the PSO a sub-routine that can be recognised and executed by functions within the SPD, and said sub-routine initiates the process of unlocking the subsequent encrypted material. Said sub-routine is encrypted using a process that is known to be reversible by functions within the SPD. The known art does not describe such a method and apparatus, which this invention references as Recursive Decryption in this application.

The decryption processes described are on the basis of encryption of information by a service provider with access to the secure information within multiple SPDs and the decryption of information in the target SPDs. PSOs may be encrypted for a specific SPD and or multiple SPDs.

The decryption processes described also may apply to the encryption of information from an SPD to a service provider. The user has no knowledge of the encryption process and usually little knowledge of the clear code being encrypted. The process can be made even more secure by the service provider sending a one off encrypted encryption process to the SPD. This process will have multiple applications and is referred to as the Coco method.

Standard Decryption and or Dynamic Decryption and or Recursive Decryption and or Realtime Decryption, and or the Coco method may be used in any PSO in any combination determined by the service provider. The service provider may always supply the required information to ensure any chosen encryption process may be reversed in one or multiple traget SPDs. The invention allows for any known method of encryption and or decryption to be used with any part or all of the invention.

The encryption/decryption methods described pertain to communications between service provider and user. They are also applicable to the secure storage of information within a UCDPS, including the encryption and storage of various values in the UCDPS memory that are intermediate and or final results of processing.

The decryption and or encryption processes described for the invention may interact in any way with external processes and the interaction may assist with said decryption and or said encryption.

The preferred security provided by an SPD is its function of decrypting and executing encrypted programs in secret and or decrypting and processing encrypted data in secret.

The invention also allows for the decryption of information that is not securely processed.

The invention allows that the SPD may be programmed with one or multiple secure user functions and any method and apparatus may be used to select the current secure user function. The system functions that perform this role are

referenced as system [redacted] switching functions and they allow that PSOs [redacted] be co-resident and or multitasking and said multitasking may occur alongside programs that do not require the use of the invention.

The use of battery backed storage elements (and or other continuous functions, e.g. security monitoring CPU) require a continuous supply of power to the device in the absence of system power. The invention allows for any method and aparatus to achieve this including the integration of a battery into the device and or an external battery together with suitably monitoring and switching circuitry. An A/D converter may be include to detect changes to battery voltage for any reason. These are referenced as secure system power management functions.

The invention as described permits:

1) the secure transfer of encrypted information from an external source (including memory) using any method, to one or multiple secure locations within a system CPU and or ESPD, and then (and or during)

2) the use of any suitable combination of microcode and or hardware and or secure internal software routines and or data (that may be augmented by any other software routines and or data in any location) securely decodes this encrypted information and or stores the decoded (and or remaining encrypted) information in a secure location (usually internal to the device, however it may include encrypted information stored in suitable external locations), and then (and or during)

3) the processing of sufficient information from the encrypted and or decrypted information (and or any other internal and or external information that is accessible, directly and or indirectly) to enable the secure and secret use of sufficient secret information that it is not practical to gain any useful benefit from any information that is in clear code and said clear code may be information that was never encrypted and or information that was encrypted and subsequently stored in unsecured locations, and

if the only reversible functional limitation applied to a software object is that which needs to be reversed by a device as described for a secret processing device, permits the original software object to be used as intended, and may do this without revealing part or all of the native object code constituting the software object, conditional upon the appropriate information being included within the SPD.

10. Automatic Reporting Facility.

A major application of the SPD as it applies to the secure distribution of software objects suitable for use on a UCDPS is to supply software objects that have been modified such that they must interact with the SPD on a frequent enough basis, that the SPD may use this interaction to record the usage of software objects, in a manner that directly and or indirectly equates to a monetary value. These modified software objects are one type of PSO as described in this application and to distinguish them from other types of PSO they are subclassified as Commercial Protected Software Objects or CPSO. A CPSO has some requirement for the exchange, directly or indirectly, of money for the use of the CPSO. The usage of CPSOs may be time and or events based and or any other method. The preferred methods allow unlimited use of these CPSOs as long as certain criteria are complied with.

As the SPD preferably does not require its host UCDPS to be attached to any remote device that may exert some form of control on the use of CPSOs and as in many instances CPSOs have no intrinsic limitation on their lifespans

and are readily available at little or no cost, a method is required to limit the use of CPSOs such that payment is made.

The invention allows for the use of CPSOs with an SPD to be controlled using any known method and apparatus and this is usually on the basis of one or multiple predefined limits electronically transferred to the SPD that are suitably adjusted as CPSOs are used. When the predefined limits are exceeded (and or in any other way reached) the SPD preferably stops processing the CPSOs. The invention allows that said predefined limits may be granted on any basis; the preferred method is to require prepayment for units. The invention does allow that there are no predefined limits on the use of CPSOs, however, this would usually only apply to major account customers and even they may prefer to have limits placed on what individual employees may spend. The SPD ensures that money is paid for use of CPSOs.

The preferred method of controlling usage of CPSOs that permit unrestricted use of the CPSO, on the basis that the SPD will record this use on any measureable units of use basis, is to prevent the SPD processing these CPSOs unless there is sufficient electronic credit within the SPD and or accessible to the SPD. This electronic credit may be stored in any form. The preferred method stores one or multiple values in the SPD.

11. An SPD may disable itself in part or whole when any requirements that are attached to the use of PSOs are not met. This includes when PSOs have been determined as being tampered with and or it is determined that an unauthorised party is attempting to use software methods to compromise the SPD and or that there is physical tampering with the SPD and or that various requirements for transferring information accumulated by the SPD directly and or indirectly have not been met and or that various electronic credits have been used and or that various keys required to activate one or multiple PSOs have not been supplied and or are incorrect and or any other reason.

12. An SPD that is disabled in part or whole may be re-enabled in part or whole by any method including the supply of an appropriately configured and validated software object.

13. Processing of Protected Software Objects by SPD: Using any suitable software routines that may be resident in the SPD and or require loading from any external sources and that may require assistance from any other SPD and or PSO and or external resources, the SPD responds to any suitable command generated by a software object requesting access to any one or multiple functions within the SPD by determining, at any appropriate stage, that a software object that has requested access to resources within the SPD is a software object that has been specially prepared to work in conjunction with the SPD and that it has not been tampered with. Such a software object said specially prepared is referred to as a PSO. A PSO is preferably encrypted, in part or whole, using any known one or multiple encryption processes. A PSO preferably includes embedded error and or validity checking information and this may use any one or multiple known methods. The process of ensuring that a software object is a valid PSO preferably includes one or multiple error and validity checking processes and the decryption and or execution of parts of the software object within the SPD.

If the object is not acceptable, the SPD may take any course of action including disabling part or all of the SPD, reporting an error to the user using any method, denying access with no report, and or any other action. An object may not be acceptable for any reason including that the object was not created for use with an SPD or that changes within the software object have occurred. If the SPD receives a predetermined number and or types of errors it may decide that these errors are not legitimate and take any course of action to protect the security of the device. This may include granting no further access and or invalidation of part or all of the secure information within the SPD. The conditions that determine this course of action may be dynamically modified by the supply of an appropriate PSO.

If it is determined that the software object is a valid software object for use with the SPD, examination of any relevant part of the software object determines what action is required of the software object. Said action may include performing further validity checking and or decryption and or any other actions as the PSO is processed in conjunction with the SPD. Protected software objects preferably include information that identifies the type of information that is included within the object, resources required of the SPD, information to assist validity and error checking of the information, information to assist decryption of encrypted information and any other relevant information. Said any other relevant information may be anything consistent with the resources of the SPD because one feature of the SPD is its capability of being securely updated to perform any software function consistent with the resources of the SPD. This updating may be dynamically performed by supplying the appropriate one or multiple PSOs prior to supplying the PSO that will use the dynamically modified functions. Said PSO that will use the dynamically modified functions may itself include in part or whole the information to said dynamically modify.

The following are the types of PSOs that an SPD suitable for use in the protection and distribution of software objects preferably includes, however, functions for one type of PSO may be combined in part or whole with any other one or multiple PSO functions to create one or multiple mixed function PSOs:

a) Secure System Update PSO: these may modify the secure system functions of the SPD using any method including data and or program instructions that are to be loaded to specific locations within secure system memory and or they may be programs and or data that is to be executed to perform one or multiple functions and or any other method. This type of PSO is preferably heavily encrypted with multiple checksums. When validated, required action is performed by the SPD.

b) Electronic Credit PSO: this adds values to one or multiple non-volatile storage locations within the SPD. Said locations are preferably clear (and or any other predetermined values) when the SPD is supplied to a user for the first time. Said non-volatile storage is preferably flash memory, described previously. Said values preferably equate to a number of units of available credit for use with various CPSOs and or any other reason. The use of these values may be for prepaid credits and these are stored in a location that is preferably decremented as available credit is used and or they may be for credits that are unpaid and are effectively a credit limit against use. Any method may be used to distinguish prepaid credits from unpaid credit.

c) Report Verification PSO: this verifies that a particular report generated previously by the SPD has been received by the SPD. It is preferably specific to a particular SPD in that unique information within the SPD is required to

correctly validate and have it perform the required functions. It may perform any one or multiple functions, directly and or indirectly within the SPD. It usually resets any restrictions within the SPD that are awaiting receipt of the report verification PSO and may do this in any way. It also usually programs the relevant locations with a new reporting interval and or modifies in any way any part or all of the report generating and verification system.

d) CPO as previously described.

Preparation of a Protected Software Object:

It is one object of the present invention to provide a method and apparatus for distributing a software object from a producer to potential users such that a user may make as many legal and or illegal copies of the software objects and distribute them as widely as they wish, however, any user executing the software object must remunerate the producer and or service provider of the software object, effectively eliminating software piracy. Part of the process to achieve this is to convert the original software object to a version that is modified to a PSO that is usually still capable of potentially running on many UCDPSs, however, those UCDPSs must be equipped with a Protected CPU, and for any particular PCPU that the PSO is to operate in conjunction with must meet the conditions of use attached to the PSO. This may or may not require intervention by the user. In following description a reference to PCPU also applies to ESPDs. The preferred method allows the user unlimited use of PSOs contingent on them having sufficient electronic credit within and or securely accessible by the PCPU The conversion from a software object to a PSO preferably occurs in a secure location.

Object Support Information:

One step in the creation of a PSO is to take a software object from the producer referenced as the primary software object and create Object Support Information (or OSI) that provides certain information to assist the execution of the PSO. The actual creation of the OSI is usually a co-operative process between the producer and service provider, however, any operations that require the use of information within the secure system memory of a PCPU would usually be restricted to the service provider. The OSI is usually placed near the start of the program, however, it may be located anywhere throughout the program as long as it is arranged in a sequence acceptable to the PCPU that will process it, and or the PSO includes various information that may permanently and or temporarily modify the PCPU such that it can locate and use the OSI. To protect the information in OSI from tampering, part or all may be encrypted, and or may have various check sums that are preferably secure and or encrypted themselves. The OSI may be provided in part or whole as a separate program(s) and or as part of one or more other programs and or may already be present in the PCPU and or any other method. If the OSI is within separate modules and contains information that the producer does not want deleted, there should be a suitably secure cross reference in the main part of the PSO to check for the presence of independent modules and valid data within. The preferred embodiment includes all information within the body of the primary software object one or multiple modules of the primary software object. The actual method to encrypt and decrypt information may use any known method and any number of levels and any combination of methods. The OSI is a description of certain functions that may be required, and they may be implemented using any known method and structure. The ability to program the secure functions within

the target PCPU enables any new structure to be created by supplying suitable PSO compatible with existing structures.

The following is a non-exclusive list of components that may be found in OSI:

Detection of Presence of a PCPU: this is usually executed immediately after the start of PSO execution. Should a PSO attempt to execute in an environment without a PCPU one or multiple adverse outcomes may result, for example the hard drive may be modified.

The preferred embodiments of a PCPU allow access to the secure memory by the execution of various special instructions. As these instructions do not exist in a normal CPU, their execution in this environment may cause problems. The preferred method of ensuring that PSOs are only used in a UCDPS that has an appropriate PCPU are:-

Common instruction trigger: a sequence of instructions that are common to a PCPU and the CPU that it replaces are executed such that a certain combination triggers various events in the secure parts of the PCPU. The following example shows one alternative:-

protected software loaded into memory

execution commences at a particular location that executes three no operation (NOP) instructions in sequence, followed by a branch to the next instruction that may be the start of three more NOPs (any number, combination and permutation of suitable instructions may be used)

the instruction following this is a branch to a routine to terminate execution of the program

a CPU that is not a PCPU will execute these instructions and quickly terminate the program

a PCPU will have the facility to recognise the particular sequence of instructions, this triggers internal routines to modify the data in the branch instruction and or redirects external execution to a particular location that enables continued processing of the PSO.

This process is transparent to the operating system.

Checking on availability of resources:

If the PSO is to execute in a multitasking environment where multiple tasks are concurrently executed on a time sliced basis, it is possible that the PCPU has a limited number of PSOs that it can handle and the next step is usually to execute a routine to determine the availability of PCPU resources and any relevant information that the PSO requires to communicate with those resources; this information may be any sort of information including a reference task number, and or an address or block of addresses the PSO should use to communicate with the PCPU, for example the user command and data ports 199 in Figure 4, and or the amount of internal PCPU memory available to the PSO and or any other information. This process may also involve the PSO providing the PCPU with certain information. In the case of the PCPU described with reference to the drawings, this transfer of information would usually be via the nominated addresses constituting the System Command and Data Ports in the dual port memory.

Should the PSO currently be unable to use the PCPU it can take any known course of action, the commonest of which may include entering a delay routine and trying again later; an efficient method is to call a routine designed

for this in the operating system, with or without a message displayed. A PCPU may have the facility to transparently override the operating system and a message may be displayed for the user to determine future action. Other actions may include program termination, with or without a message.

A PSO preferably checks various information currently resident within the secure system memory of the PCPU for the presence of certain functions within the system memory and that they are a version suitable for use by the PSO. This is usually confirmed by checking that the current version number of system memory functions are current for a particular PSO, however, it may use any method. Should certain functions not be current, the invention allows that a PSO may be shipped with certain update information included as part of the PSO and or with other PSOs shipped with the PSO, and that a PSO may automatically and or at the users direction, update the system memory functions to current information and may suitably adjust the version number, and that this may be a temporary modification for the duration of execution of the PSO and or a semi-permanent and or permanent change. Should the system functions not be able to be updated for any reason, the PSO would usually terminate with a request for the user to arrange for the necessary changes to system functions, however, it may take any other action.

Conditions of Use:

As PSOs may need to identify to the PCPU the producer of the PSO (e.g. to log usage and allocate payments), a unique vendor identity code may be included in the PSO in a position and or any other way that can be determined by the PCPU. This code is usually consistent on each product from the producer. The invention allows for this method or any other to differentiate PSOs that are primarily commercial objects from those that provide various support functions.

To differentiate a particular program from others by the same producer a unique program identity code (UPID) is usually included in the PSO in a known location and or any other way that can be determined by the PCPU. This may be unique amongst products from the same producer, however, it may be identical to another product by another producer. This code may be further used to categorise a particular program e.g. part of the code may identify the program as a game or a wordprocessor, etc., and this would usually be common across all UPIDs, another part may identify the version number and the balance may be used to ensure that the UPID is unique to any others from that producer. Any other relevant information may also be included in the code. The invention allows that the various sub-parts of information included in this code may in part or whole be allocated their own codes.

The invention allows that the billing for the use of a PSO may use information included within the PSO. Any of the following information may be located where the PCPU and or any other applicable devices or routines can identify it:

Currency Identifier - this indicates the currency in which the producer of the PSO is to be paid. It is mainly used by the service provider, however, it may be used for any reason.

the target PCPU enables any new structure to be created by supplying a suitable PSO compatible with existing structures.

The following is a non-exclusive list of components that may be found in OSI:

Detection of Presence of a PCPU: this is usually executed immediately after the start of PSO execution. Should a PSO attempt to execute in an environment without a PCPU one or multiple adverse outcomes may result, for example the hard drive may be modified.

The preferred embodiments of a PCPU allow access to the secure memory by the execution of various special instructions. As these instructions do not exist in a normal CPU, their execution in this environment may cause problems. The preferred method of ensuring that PSOs are only used in a UCDPS that has an appropriate PCPU are:-

Common instruction trigger: a sequence of instructions that are common to a PCPU and the CPU that it replaces are executed such that a certain combination triggers various events in the secure parts of the PCPU. The following example shows one alternative:-

protected software loaded into memory

execution commences at a particular location that executes three no operation (NOP) instructions in sequence, followed by a branch to the next instruction that may be the start of three more NOPs (any number, combination and permutation of suitable instructions may be used)

the instruction following this is a branch to a routine to terminate execution of the program

a CPU that is not a PCPU will execute these instructions and quickly terminate the program

a PCPU will have the facility to recognise the particular sequence of instructions, this triggers internal routines to modify the data in the branch instruction and or redirects external execution to a particular location that enables continued processing of the PSO.

This process is transparent to the operating system.

Checking on availability of resources:

If the PSO is to execute in a multitasking environment where multiple tasks are concurrently executed on a time sliced basis, it is possible that the PCPU has a limited number of PSOs that it can handle and the next step is usually to execute a routine to determine the availability of PCPU resources and any relevant information that the PSO requires to communicate with those resources; this information may be any sort of information including a reference task number, and or an address or block of addresses the PSO should use to communicate with the PCPU, for example the user command and data ports 199 in Figure 4, and or the amount of internal PCPU memory available to the PSO and or any other information. This process may also involve the PSO providing the PCPU with certain information. In the case of the PCPU described with reference to the drawings, this transfer of information would usually be via the nominated addresses constituting the System Command and Data Ports in the dual port memory.

Should the PSO currently be unable to use the PCPU it can take any known course of action, the commonest of which may include entering a delay routine and trying again later; an efficient method is to call a routine designed

for this in the operating system, with or without a message displayed. A PCPU may have the facility to transparently override the operating system and a message may be displayed for the user to determine future action. Other actions may include program termination, with or without a message.

A PSO preferably checks various information currently resident within the secure system memory of the PCPU for the presence of certain functions within the system memory and that they are a version suitable for use by the PSO. This is usually confirmed by checking that the current version number of system memory functions are current for a particular PSO, however, it may use any method. Should certain functions not be current, the invention allows that a PSO may be shipped with certain update information included as part of the PSO and or with other PSOs shipped with the PSO, and that a PSO may automatically and or at the users direction, update the system memory functions to current information and may suitably adjust the version number, and that this may be a temporary modification for the duration of execution of the PSO and or a semi-permanent and or permanent change. Should the system functions not be able to be updated for any reason, the PSO would usually terminate with a request for the user to arrange for the necessary changes to system functions, however, it may take any other action.

Conditions of Use:

As PSOs may need to identify to the PCPU the producer of the PSO (e.g. to log usage and allocate payments), a unique vendor identity code may be included in the PSO in a position and or any other way that can be determined by the PCPU. This code is usually consistent on each product from the producer. The invention allows for this method or any other to differentiate PSOs that are primarily commercial objects from those that provide various support functions.

To differentiate a particular program from others by the same producer a unique program identity code (UPID) is usually included in the PSO in a known location and or any other way that can be determined by the PCPU. This may be unique amongst products from the same producer, however, it may be identical to another product by another producer. This code may be further used to categorise a particular program e.g. part of the code may identify the program as a game or a wordprocessor, etc., and this would usually be common across all UPIDs, another part may identify the version number and the balance may be used to ensure that the UPID is unique to any others from that producer. Any other relevant information may also be included in the code. The invention allows that the various sub-parts of information included in this code may in part or whole be allocated their own codes.

The invention allows that the billing for the use of a PSO may use information included within the PSO. Any of the following information may be located where the PCPU and or any other applicable devices or routines can identify it:

Currency Identifier - this indicates the currency in which the producer of the PSO is to be paid. It is mainly used by the service provider, however, it may be used for any reason.

Personal User Device Valid - this indicates whether this PSO may be used with a Personal Software Card. This is a device described in another application that lets the users of one UCDPS temporarily or permanently port various access and billing to another UCDPS.

Timed Basic Charge (or TBC) - is the unit rate for use of the product. The preferred rate is by the hour, however, any time interval may be used.It is anticipated that users will ultimately determine the type of billing they want, and it will probably be based on a time used basis associated with certain frequency discounts and possibly a cut off point at which there are no additional charges. The charge rate is usually in terms of a standard unit - for example it may be US Dollars. Whatever standard rate is chosen is usually standardised across PSOs. The invention allows that any amount in any currency may be used. The invention also allows that the TBC for various countries may be different, for example to allow for different economic conditions. Any particular PSO may include the entire set of TBCs for all countries or only a subset. The TBC may not be available to all regionals. The invention allows that a discount schedule may apply to the TBC for increasing use or whatever reason, and that this may vary from one region to another, and this discount schedule may be stored in the PSO. Further discounting may apply for different types of users, e.g. government, education, business and part or all of this information may be stored in a PSO. Various vendors may wish to offer various discounts for existing customers when an updated version of their product is released and or when a new product is released and these may be stored in a PSO.

The PSO usually includes one or multiple transaction processing codes to indicate the type of billing system used. This may vary from region to region and each PSO may have a list that includes transaction processing codes for all countries or any subset. For any particular country, there may be different codes for different groups eg, government users may be billed using a different method to business, and the combinations used may vary from one region to another.

While not an exclusive list, the following are the more common types of transaction processing codes:-

a) The PSO may be distributed at nominal cost, with the customer paying for time used.

b) The PSO may be distributed at nominal cost, with the customer paying for time used, however, a data key (at no cost) is required to activate the program.

c) The PSO may be distributed at nominal cost, with the customer paying for time used, however, a data key is required to activate the program and there is a charge for the key; this charge may be located in the relevant fixed basic charge field.

d) The PSO may be distributed at nominal cost, however, a data key is required to activate the program and there is a charge for the key, however, there are no continuing charges.

e) The PSO is only supplied on receipt of payment, with additional charges for time used. A key may be required to activate the program.

f) The PSO is only supplied on receipt of payment, however, there are no additional charge.

The PSO may be one that is generic to multiple PCPUs or customised to a particular PCPU.

Event Basic Charge (or EBC) - the invention allows that usage of software may be based on the number of times the program is opened and or any other event based mechanism. The Event Based Charge is the unit rate for this method

of billing. All of the options and or discounts and or requirements described for TBC above apply for Event Based Charge and will not be repeated, however, the various combinations and particular options used may vary from the TBC in any way.

Fixed Basic Charge (or FBC) - this is a fixed charge to use the software and may be a one off charge that subsequently permits unlimited access on that UCDPS or a charge that grants access and then bills on a usage basis using any combination of the previous methods. All of the options and or discounts and or requirements described for TBC above may be applicable for Fixed Basic Charges, however, the various combinations and particular options used may vary from TBC in any way.

Transaction processing codes may be constructed to detail any combination of billing processes and discounts and anything else.

The ability to distribute software in massive quantities with very low upfront costs to the user may provide significant changes to the methods of marketing and advertising software products. One method may be to permit the user free or discounted access to various products, particularly new products. This may include various promotional schedule codes (PSC) within the PSO, that may be designed to achieve any outcome that is permitted by the PCPU, that the PSO executes on, and this may include codes representing anything to do with promoting any sort of product using any known method, including:-

- a list of discounts and the time they apply may be included within the PSO, and they may be multiple. The discounts may be any value, and may result in free software for variable periods of time. The facility even exists for a producer to pay a user to try their product. Particular promotions may have a use by date attached to them.
- Another approach may be to generate a random number in the PCPU each time a program is initiated or on any other basis. If this matches a code in the PSO, then various free program time may be provided on the current PSO and or another program by the producer and or various prizes may be given away.
- The software may also be made available to a potential user with part of its functions disabled, and no charge or a nominal charge applied to the use of this partially disabled program. This may be particularly useful for programs that may take time to assess, for example a new accounting program, where a potential customer may want to fully assess the package prior to committing to a changeover from an existing system. The activation to a fully operational system may require a key (that may or may not have a charge) or simply require the user to execute a program that initiates time and or event based billing, or any other method.

The information to perform any promotional function may be included in part or whole within the PSO, however, it would usually rely in part or whole on secret processes within the PCPU to prevent unauthorised manipulation of the promotions.

Certain software products may be unsuitable for use by particular groups. For example, certain countries may be restricted from using software because of security concerns and or because it may offend certain cultures and or other software may be unsuitable for children and or it may be restricted to certain professions and or it may be

restricted to use at certain times and or for any other reason. These are referenced as Group Restriction Codes (GRC) and may be included in a particular PSO to limit access to various categories of user.

Any information included in a particular OSI may become obsolete and this may be a particular problem with prices and discounts. Any information contained in a OSI may be replaced in part or whole with other more readily updated information stored in any suitable location; this may include locations within the PCPU, and or various files stored on one or multiple mass storage devices, and or distributed with other PSOs, and or distributed as part of codes supplied to users to update PCPU credits and or any other reason, and or any other method. All of this may be subject to the overall control of the service provider who can vary the actual amount charged to any particular user. The billing process is described later in this application.

Part or all of the information within the OSI is usually reliant on known information within the secure system memory of the PCPU to correctly interpret and or execute the various functions, however, as part or all of this PCPU memory may be reprogrammed by suitably encrypted external information, part or all of which may be included within the PSO, the specific requirements of a particular PSO may be met by dynamically modifying part or all of the secure system memory. Additional flexibility may be gained by loading any required part of the PSO into secure user memory for execution. Although various functions have been detailed for the OSI, in practice a multiplicity of special functions may be included and these may occur during any part of the execution of the PSO.

Method to update the PCPU:

Another step in the preparation of a PSO may be to include in the PSO various routines and data that will execute automatically and or under user control to update various information on the UCDPS for any reason and may include:-

- update the secure system memory
- update various files stored on a UCDPS that contain various billing information and or discounts and or special promotions and or any other information.

These update functions may be included as part of the actual PSO and or as part of one or more other PSOs. These other PSOs may be created specifically for the purpose and or may be parts of other PSO applications. These other PSOs may be supplied to the user with the said actual PSO and or may be supplied separately.

Error and Validity Checking:

A PSO, and the PCPU with which it is to operate, are provided with a number of secure mechanisms to protect against unauthorised analysis of information stored within. As there may be considerable financial gain to any party that manages to compromise the security of either, it is anticipated that a number of attempts will be made to compromise the security of both, and one method may be aimed at changing various parts of the PSO in an attempt to analyse the various outcomes. In order to protect against this and also to detect genuine errors in the PSO, it is usual to use one or more error and or validity checking processes on information within the PSO, and these may use any known method and apparatus, and these may be dependent in part or whole on functions within the PCPU, that may include:-

- routines within system memory, and or

- various algorithms implemented in hardware within the PCPU, and or
- routines loaded from external sources (usually, in part or whole, in encrypted format), and or
- loaded from the PSO (usually, in part or whole, in encrypted format), and or
- any other source.

The error checking and validity checking is a process that usually occurs in total secrecy at both ends, with the service provider the only party that knows the process. The service provider is aware of the processes available in any particular PCPU to extract and validate any parity information and or CRC information and or any other information, and the method used to take the actual code of the PSO and generate the expected parity information and CRC information and any other information, and the methods to determine whether or not the expected information matches the extracted information. The service provider can take a PSO at any stage or stages in the conversion process from software object to PSO and analyse the information and add and or change data in such a manner that the outcome when run through the error and validity checking process in the PCPU will not detect any errors. Should one or multiple parts of the PSO be changed by an unauthorised party, then the error and or validity checking process in the PCPU will detect the modifications and may take any known action, including those actions described later. If the service provider prepares a PSO for error and validity checks and the process complements a protocol preprogrammed into the PCPU, there may be no need for any other additional information within the SPO, however, if the service provider follows a variable pattern and or non-standard processes then additional information may need to be included within the PSO to permit correct analysis at the other end, and this may use any known method. As part or all of the PSO will usually be subsequently encrypted, there is no practical way for an external analysis of the PSO to even hint at which apparently meaningless data is part of error/validity checking and which is encrypted information. Furthermore, the error/validity checking information may itself be encrypted. Furthermore the system usually only needs to work in one direction - provider to user, although some processes may need to be included within the PCPU to generate error and or validity checks on information that is to be stored in encrypted format in external resources (these are discussed in more detail in the applications dealing with these devices). Any number of error detection and validity checking processes may be applied and these may occur during various levels of the encryption process. The invention also allows that error and or validity checking may be performed on part or all of the PSO with the actual method to reverse this included within the PSO, and as long as part or all of the method to reverse is encrypted and the reversal process occurs in secrecy, there is no means of reverse engineering the process, and the actual methods and or apparatus used may be any known method and or apparatus.

Encryption of the information to create the Protected Software Object:

The final step in the creation of a PSO is the conversion of the software object as supplied by the producer together with any additional information as previously discussed to a protected program that provides the security against illegal use of the program. By encrypting the PSO using any known encryption method and any combination of known encryption methods, including the processes described previously, the software object is converted to a PSO that in part or whole may only be executed internal to an appropriate PCPU. The software object may be encoded to one and or multiple levels of complexity. The software object is preferably analysed to determine which parts require encryption, what method or methods of encryption should be applied and any ancillary information that is required to support these methods. The actual arrangement of information within any part of the PSO to effect various

outcomes will be highly variable with the exception of certain functions fixed by a particular PCPU, and as the present invention allows for the provider supplied PSO to be flexible and the functions within a particular PCPU to be programmed in a multiplicity of ways, the various combinations and permutations to achieve the same outcome are obvious, once the specific requirements and one method of achieving this are described.

Crediting funds into a PCPU (and or other PCPU):

The present invention allows that a part of the secure system memory of a PCPU may be securely programmed with information that indicates an amount of credit (using any method and or currency) that may be offset against software usage (and or any other applicable uses). Various secure locations within the PCPU within a particular UCDPS may contain codes that are unique to that particular PCPU and these codes are usually secret. A particular PCPU usually has a publicly accessible electronic signature that can be used to identify a particular UCDPS. A particular PCPU may also have other characteristics that are unique to a particular PCPU, for example, particular software routines and or encryption/decryption processes and or any other applicable variation. Because of the secure nature of information contained within a PCPU, it is preferable that conversion of a software object into a PSO is performed by a service provider, and that the actual information within PCPUs is maintained in a secure environment. When a UCDPS is initially shipped to a customer, it is likely that the PCPU has no credit value programmed within and may not be activated to execute PSOs. The process of activating a particular PCPU may be accomplished by any method and apparatus, including:

1) The user contacts a service provider (using any method, the most convenient usually being via a modem) and supplies the service provider with the serial number of the PCPU, the amount of credit required, and payment details (that is preferably a credit card payment) that may use any known method.

2) Using known details about various information within that particular PCPU, the service provider uses the requested amount of credit and encrypts this amount using any known method and apparatus (and an experienced person should be able to devise multiple techniques based on the encryption/decryption processes described earlier). The encryption process that may use any information (including time and or date and or any other unique and or global information within the PCPU and or that may be securely transferred to the PCPU, using any known method including those described in this application) to generates a one time code that may be decrypted within the PCPU.

3) The one time code is transferred to the user of the PCPU and entered into the computer. The code is decrypted. If an error is generated, the user may be advised. Once the amount is confirmed the nominated credit is programmed into any appropriate secure non-volatile location internal to the PCPU that cannot be tampered with.

4) This process may activate the PCPU if required, however, the preferred determinant as to whether or not a particular PCPU will execute one or multiple PSOs is based on the amount of available credit.

5) The available credit is progressively decremented as various PSOs are used, and the present invention allows for any method and apparatus for billing for PSO use.

6) Software usage of various software objects may be logged. This is described later.

7) When the credit amount is decremented to a predetermined amount (and said predetermined may be by the service provider and or the user) the user is advised that additional credit will be required shortly. The method of advising the user of an imminent shortage of credit may use any method and or apparatus, however, as the programs that implement this process are preferably executing in part or whole from within secure memory internal to the PCPU, the facility exists to generate an internal interrupt and jump to an appropriate internal and or external

program. This may occur at any time, with the most usual being shortly after a system reset. The process may be transparent to the operating system. The facility exists, using a similar process (and or any other method and or apparatus) for the user to generate a current report of available credit and or software object use.

8) For the second and subsequent contacts with a service provider to refresh the credit available within the PCPU, in addition to providing the service provider with the electronic signature of their PCPU, the user will usually be required to advise the service provider of a code (that is securely generated within the PCPU using any known method and apparatus within the PCPU) that may include current information on remaining credit (that may be zero) and may include information on the usage of part or all software objects that have been used in the period.

9) Step 2 is repeated, however, in addition to credit information, the code supplied to the user usually contains an encrypted message that informs one or multiple routines within the PCPU that information pertaining to software object use has been received by the service provider. Storage locations allocated to this information may then be cleared.

The present invention allows that although the process as described requires prepayment for services, the process is also compatible with the provision of credit within the PCPU on account terms with selected users, and the credit amount allocated would usually be sufficient to cover expected usage (or may be any amount). The actual amount to bill the user may be calculated by subtracting the amount of credit remaining from the amount supplied in the previous period and or any other method and apparatus.

A user friendly menu system may be used to assist part or all of the process described above.

Monitoring the use of protected software objects:

The present invention allows for any known method and apparatus that can monitor and or record the usage of PSOs (and or software objects), and preferably one that is compatible with multitasking programs in a single processor and or multiprocessor environment, and preferably one that provides a tamperproof, secure system that operates in part or whole from within a PCPU and or any other SPD, when the UCDPS is an independent entity, and or when independent and connected to a network and or when independent and connected to Internet or similar, for its correct functioning, and or when the UCDPS is dependent in part or whole on connection to a network, and or is dependent in part or whole on connection to the Internet (or similar). In a single task UCDPS the SPD usualy starts recording usage when activated and terminates when the PSO finishes. The preferred method in a multitasking environment where usage is timed is to generate an internal interrupt within secure microprocessor on a periodic basis, and said interrupt activates a routine within internal secure memory that retrieves the contents of the program counter of the system microprocessor at the time of the interrupt and compare this with an address map generated by the PSO to determines which program was executing during the interrupt. The invention allows for any combination and or permutation and or weighting for usage of any one or multiple PSOs. Event usage may only require counting ocurrences of the measured event in single and multitasking UCDPS. The usage of PSOs is usually recorded in part or whole within secure internal memory, however, the invention allows that part or all of the information on the use of PSOs may be encrypted and stored external to the PCPU and or UCDPS. It is preferable to keep sufficient information on PSO use internal to the device, in order that a software vendor receives the appropriate payment in the event that external storage of this information is corrupted, in which case while there may be no detailed

basis, and said period basis may be any period and it may be varied by the service provider, and or the user may be required to report to the service provider when a certain number of events have occurred, that may be any combination of events, including the number of times one or multiple PSOs have been used, and or a user may be required to provide a report to any authorised party for any reason; those PSOs that do not require the presence of available credit within the SPD may share any of the reporting requirements discussed, however, they usually are independent as to the state of credit within the SPD. In practice a mix of methods may be used and a periodic report may be required. When a report is required on a periodic basis, a secure battery backed realtime clock/calendar is the preferred source of determining (in conjunction with predetermined and or otherwise information on the time intervals to be used) when the relevant time interval has occurred. When available credit expires and or a certain date and or time is reached and or a certain number of events and or type of events have occurred, part or all of the functions of the SPD may be disabled.

Whatever the trigger point for requiring the user to supply the service provider with a report generated by secure methods within and or in conjunction with the SPD, the method steps to supply said report and to reactivate the SPD for further use may use any method and apparatus, including:

1) When the SPD determines that internal and or external information is due for reporting to a service provider, any method may be used to alert the user, and one preferred method uses the ability of the PCPU to call routines transparently to the operating system by having the secure microprocessor DMA information to display memory and this facility may be used to overlay a message on the display device of the UCDPS advising them to execute a program that will generate a report and this is preferably at the start of a processing session.

2) The report generator is executed and this may display a menu based system to assist the user through the process. If information is to be transmitted to the service provider via a modem and any return information received by the same method then the process may be fully automated and transparent to the user. The invention allows for any method and apparatus that assists the user with the process. The report generator usually triggers routines within the SPD that collate and encrypt the information to be supplied to the service provider, with the information usually including one or multiple unique identity codes for a particular SPD, and this may and or may not be encrypted. The report would usually be integrated with any information to be supplied to a service provider as regards credit remaining within a SPD.

3) The user contacts a service provider (using any method, the most convenient usually being via a modem) and supplies the service provider with the information generated by the report generator. As mentioned, if using a modem this process may have minimal user intervention. If a modem is not available the information may be sent by any method, including as a file on a diskette and or the information may be read over a telephone (this may be verbal or use the numeric pad) and or any other method.

4) On receipt of the information the service provider determines the electronic signature of the SPD generating the report and using known details about various information within that particular SPD decrypts the report and confirms that it has not been tampered with.

5) Any method may be used to collect payment for any amounts payable as a result of use of software objects and or any other reason.

6) The service provider prepares a one time code using any method and apparatus that may be correctly interpreted by the target SPD and is usually specific to a particular SPD.

7) The one time code is transferred to the user of the SPD and entered into the computer. The code is decrypted. If an error is generated the user may be advised. The purpose of this information is usually to advise routines within the SPD that a correctly encoded report was received by the service provider and that further use of PSOs may proceed. Other information, e.g. credit may be included with said one time code. The normal process preferably provides a report and ensure continued use of PSOs prior to the expiry date of the current period.

With the exception of the periodic updating of internal credits and the reporting of software usage the method and apparatus of software protection and distribution may be transparent to the user. As long as payments are made as required the user would treat a PSO as they would any presently available software object.

The invention allows that a user may purchase a particular PSO for unlimited use, and this may use any method and apparatus, including debiting the cost of the PSO from any available internal credit and setting a code such that there is no further billing for use of this PSO; one method allows for a file to be kept on a suitable mass storage device attached directly and or indirectly to the UCDPS (referenced as Exempt PSO File, or EPF) and this may store, usually in encrypted format (in part or whole), a vendor code and product code and a code that is unique to a particulrar PCPU for that particular product. Said code is usually created when payment is made and this may be automatic when there is available credit in the PCPU and or may be supplied by the service provider on receipt of payment and or any other method. When a PSO is loaded for execution, routines within the PCPU may access this file and determine whether or not a particular PSO that is normally charged on any type of usage basis, is exempt from this process. One alternative is for the service provider to credit any debits made for use of prepaid PSOs.

A variation on the method and apparatus described earlier allows for a certain group of programs to be used on an unlimited basis for a period of time, for one fixed charge. This may apply to computer games for example that may be used for $X per month, where X may be any amount. A periodic report is required to determine usage of the different games in order to appropriately pay the vendors of those games. The actual pro rata allocation to various vendors may be made by the service provider using any agreed formular. This may use a special code within the PSO and or the CDF and or the EPF and or any other method. The invention allows that multiple software object groupings may use this variation and the amount charged for one grouping may be the same and or different to other groupings.

The invention allows that part or all of the processes that require the user to supply one or multiple codes to activate part or all of the invention for any reason, may use any method and apparatus to prevent attempts at creating said codes by trial and error and or any other method, with the preferred said method and apparatus to prevent, being a routine(s) within secure internal memory that log in non-volatile storage invalid attempts at entering codes and part or all of this information may be stored in one or multiple external files, that may be directly and or indirectly attached to the UCDPS. The invention allows for any action to be taken including, disabling the PSO and or multiple PSOs and or the PCPU and or all processing capability, and this may be done using any method and apparatus.

The invention allows that a user who has purchased in part or whole one or multiple PSOs and or earned frequency discounts on one or multiple PSOs and or any other reason, may wish to port these to another SPD for any reason, including that the user has purchased a new machine and or because the user wishes to sell part or all of any interest in one or multiple PSOs to another user. The invention also allows that one or multiple PSOs may not offer this facility. The invention allows that there are multiple known methods and apparatus for achieving this including, the preferred option that may involve the following method steps:

1) the user activates a program to reverse various capabilities granted to a particular SPD, for example activation codes and or discount schedules. This would usually initiate a menu type screen on the display device, using the method previously described, of the UCDPS to assist the process.

2) the user nominates those PSOs that are to have part or all rights of use transferred to another SPD.

3) the program may change various internal locations and may change various external locations such that existing rights are no longer valid on the SPD.

4) encrypted information is supplied to the service provider indicating that various access rights to one or multiple PSOs have been modified, and the encrypted information (using any method and apparatus) is decrypted and verified for validity, using any method and or apparatus.

5) the user usually informs the service provider of the new SPD that various access rights are to be transferred to. This may be multiple SPDs.

6) any codes and or discounts and or new versions of encrypted PSOs are prepared for the nominated PSOs and supplied accordingly.

User Password:

Certain information is preprogrammed into the PCPU prior to being made available to a user and some of this may restrict the user of that particular PCPU from various functions available within the PCPU and or available in various information supplied by a service provider. An example may to restrict users of a particular country from various services. The invention allows that some of these restrictions may be reprogrammable with information supplied by the service provider while other information may be fixed. A user of a UCDPS equipped with a PCPU may have various restrictions that they want placed on the use of the PCPU and these would normally be programmable by the user, and these may included any approved functions, using any known method. A user may want a master password for themselves and this would usually be stored within non-volatile storage elements of system memory, and the correct entry of this may be required to activate the PCPU (in the case of a PCPU the CPUs within may be disabled). Additional passwords may also be required that allow limited access to the PCPU, for example, certain passwords may be attached to children to prevent them from using unsuitable software, or certain employees may be prevented from playing games on their computers during business hours. Certain functions may also be attached to various passwords, e.g. to monitor usage.

Any program and or data that is preprogrammed into a PCPU may in part or whole be the same as those within other PCPUs and or may in part or whole be unique to other PCPUs. Any program that is currently within secure memory may call on any currently external programs and or data and or apparatus to assist the functions of said any program.

Protection of other forms of information:

The present invention also allows for the inclusion of part or all of the method and apparatus described in this application when used in conjunction (in any manner) with any secure apparatus (that may be one or multiple devices) for use in:

the secure decoding of encrypted (in part or whole) video information and or any other encrypted (in part or whole) visual information, and or the secure generation of the necessary signals to display the decoded information on a suitable visual output device, with said necessary signals preferably constrained within a secure location within said visual output device and or

the secure decoding of encrypted (in part or whole) sound information and or the secure creation from this decoded information of the necessary signals to drive a loudspeaker (and or equivalent), with said necessary signals preferably constrained within said loudspeaker (or equivalent) and or

the secure decoding of encrypted (in part or whole) text as may be the case with electronic books and or newspapers (and or any other printed matter of commercial value that is published in electronic form) and the secure generation of the necessary signals to display the decoded information on a suitable visual output device;

this particularly applies when said secure apparatus securely monitors and or logs (directly and or indirectly) the use of the encrypted information as it is decoded and used within said secure apparatus, and or

that includes (directly and or indirectly) one or multiple methods and apparatus to ensure payment is made for said use.

Any combination of software and or hardware and or microcode may be used to implement the method and apparatus, with the preferred method and apparatus:

retrieving pricing information from the encrypted information; and or

timing the use (and or counting the frequency of use) of said encrypted information; and or

storing this within the secure apparatus (that may include secure locations external to the secure apparatus) in non-volatile storage elements; and or

debiting an amount of electronic funds previously embedded within the secure apparatus; and or

recording an amount to charge at a future date; and or

generating a report of usage (preferably with a breakdown for each vendor and or product) that is supplied to the information provider (and or agent); and or a

system to ensure that said report of usage has been received by the relevant parties; and or

that may disable part or all of its capabilities in the event that electronic funds expire and or internal credit limits are exceeded and or a report is not provided to the relevant parties and or that periodic information is not received from said relevant parties; and or

that may be updated with additional electronic funds and or any previously used (or expired) credit limits reset. The encrypted information may be supplied on any machine readable physical media (e.g. CDROM or Videodisc) and or broadcast using any method.

When an external PSO requires to access the SPD, the normal process is to:

a) block interrupts if required and write a command to the system command input port requesting use of the SPD.

b) the process of writing to the port preferably generates an interrupt so there is a rapid response from the secure microprocessor, otherwise there may be a delay while it is polled.

c) the secure microprocessor writes to the system command output port a value that indicates if there are currently no resources and another value if there are resources, together with the address and size of a user command input and output port and a user data input and output port. It clears the value written by the system microprocessor into the system command input port.

d) the PSO reads the information from the system command output port and reactivates interrupts.

e) if resources are currently unavailable to the PSO it may enter any known delay routine and try again later. The option exists for it to branch to a routine to advise the user that the multitasking capability of the UCDPS is currently fully extended.

f) if granted access it saves the appropriate user port information in an accessible location and may read and write to these ports as required. There is no need to disable interrupts when accessing the user ports allocated to it. There is no requirement to modify the task switching routines of the UCDPS operating system.

g) if the SPD has granted a PSO access to the SPD then it preferably stores relevant information about the PSO user partition in a known location in the system partition, usually with information on other user partitions.

h) the SPD waits until the PSO starts writing information to its user data input port, this may be triggered by an interrupt or polling of locations and or any other method.

i) the SPD transfers the information into the allocated secure user partition. This may be done via the user data input port and or via Direct Memory Access (DMA) or by direct programmed I/O by the secure microprocessor and or any other method permitted by a particular embodiment of the invention.

j) PSOs usually include various information to assist the SPD in addition to various encryption and validity checking information.

k) various system functions are activated to decrypt and validate where appropriate and extract other information relevant to the PSO.

m) the PSO may be determined to be a valid System Support Object that is required to be loaded into the secure system partition to addresses determined by any method. The system Support Object may include data and commands as to what sort of processing is required and or it may contain executable instructions, in which case the secure microprocessor will be directed to execute this program.

This is usually granted if the SPD currently has sufficient resources. This would normally be the case in a single tasking system, however, in a multitasking environment, an PSO may need to wait. Said wait may use any method.

The claims defining the invention are as follows:

1. A method of distributing software objects from a producer to a potential user comprising the method steps of:

equipping a user controlled data processing system with a secret processing device, and said user controlled data processing system equipped with said secret processing device is referred to as a PUCDPS, wherein said secret processing device of said PUCDPS may be configured to be dependent in part or whole on the coupling of said PUCDPS for part or all of the time, to one or multiple remote computers and or any other data processing devices, however, part or all of said secret processing device may operate and or be configured to operate in a stand alone PUCDPS and may remain operational for extended periods after said PUCDPS is removed from a source of power one or multiple times, and or moved to different locations, and or reset one or multiple times, and or any other event that would normally disrupt processing on said PUCDPS;

providing one or multiple service providers, with part at least of secret information within one or multiple said secret processing device that is required to provide part at least of the services required by one or multiple said PUCDPS, wherein said service providers are the agents of said producer;

providing a software object;

modifying part or all of said software object such that it is functionally limited to require said PUCDPS for correct processing (in this claim execution and process and processing are interchangeable and refer to execution of instructions and or processing of data) and the functional limitation may be Oscar compatible and or may be Groover compatible and or may use any encryption method able to be reversed in said secret processing device, furthermore, said functional limitation may be of one or multiple essential parts of the software object such that it is *not practical* to regenerate the original software object from any parts that are not functionally limited, and for any particular functionally limited software object the functional limitation may only be reversed in part or whole by a specific said secret processing device with unique characteristics necessary to reverse the functional limitation, or the functional limitation may be reversed in part or whole on a plurality of said secret processing device identified by common characteristics necessary to reverse the functional limitation; and or

modifying part or all of said software object, using any method, such that said software object is securely linked in part or whole, using any method, to any one or multiple conditions of use, that in part or whole are not practical to tamper with and said conditions of use may include any code that identifies the producer of said software object and or identifies said software object in any way, such that when said secret processing device is used to reverse part or all of said functional limitation, said secret processing device may record use of said software object and or the use of software objects of a particular producer and or any other record that in part or whole is used in determining remuneration to the producer and or any other parties and or said conditions of use includes any code that contains information which may be used by the SPD to determine if said software object:

is permitted to execute and or process in part or whole on a units of time used basis, and may include what fee should be applied for the use of said software object and said fee may be any unit of measurement and is preferably a generic units of use basis and said generic units may be attributed any real currency value at any stage; and or

is permitted to execute and or process in part or whole on an events occurring basis, for example the number of times one or multiple parts of said software object are loaded and or executed and or any other measurable events basis, and may include what fee should be applied for the use of said software object and said fee may be any unit of measurement and is preferably a generic units of use basis and said generic units may be attributed any real currency value at any stage; and or

is permitted to execute and or process on an unlimited basis subject to a fee, and may include what fee should be applied for the use of said software object and said fee may be any unit of measurement and is preferably a generic units of use basis and said generic units may be attributed any real currency value at any stage; and or

is permitted to execute and or process on any type of limited basis subject to a fee, and may include what fee should be applied for the use of said software object and said fee may be any unit of measurement and is preferably a generic units of use basis and said generic units may be attributed any real currency value at any stage; and or

requires entry of one or multiple data keys of any type prior to initiating use of part or all of said software object for the first and or any other time on a particular said secret processing device and may include whether or not a fee is to be charged; and or

requires any other restrictions of any type to be placed on use of said software object ; and

any said software object modified in part or whole as described is referred to as a Protected Software Object;

providing one or multiple protected software object onto computer-accessible memory media and or any suitable apparatus for electronically transferring said protected software object to a potential user, and preferably the conditions of use attached to said one or multiple protected software object permit said protected software object to be used on a time used basis in a PUCDPS with a secret processing device that has sufficient quantity of one or multiple said unit of measurement stored within and or securely accessible;

shipping said one or multiple said protected software object on said computer-accessible memory media to a potential user and or said electronically transferring said one or multiple protected software object;

loading said one or multiple said protected software object into said PUCDPS and executing as permitted by said conditions of use;

where required by said conditions of use, a user friendly menu system and or any other method provides for the user to:

request the supply of one or multiple said unit of measurement that may be required by the said secret processing device for any purpose, and or

receive one or multiple said unit of measurement, preferably in suitably encrypted format, that may use any method, and transfer said unit of measurement into the said secret processing device, and or accessible to the secret processing device, and or

request the supply of one or multiple data keys that may be required by the said secret processing device, and or

receive one or multiple data keys and transfer said data keys into the said secret processing device, and or accessible to said secret processing device , using any method, and or

generate one or multiple reports of software usage and or any other information that may be required, and supply said reports to said service provider and or any other external location, as required, and or

receive one or multiple codes confirming that said report has been received and supply said one or multiple codes confirming into said secret processing device and or accessible to said secret processing device, and or

request the service provider and or any other authorised party for one or multiple codes that may be used to reactivate part or all of said secret processing device that may have been disabled for any reason, and or

receive one or multiple codes to reactivate part or all of said secret processing device that may have been disabled for any reason and transfer said codes into said secret processing device, and or accessible to said secret processing device, and or

for any of the preceding, the information generated by said PUCDPS and or received from said service provider is preferably transferred electronically, however, any other combination of methods may be used including mailing of computer-accessible memory media containing the information.

2. A method of distributing software objects according to Claim 1, wherein said secret processing device may:

securely decrypt and execute (in this claim execution and process and processing are interchangeable and refer to execution of instructions and or processing of data) and or process instructions and or securely decrypt and process data; and or

securely decrypt and execute and or process instructions and or securely decrypt and process data that complies with part or all of the requirements of reversing functional limitations applied that are said Oscar compatible; and or

reverse any functional limitations applied that are said Groover compatible; and or

reverse part or all any functional limitations applying to said protected software object; and or

may decide to reverse one or multiple said functional limitations applied to one or multiple said protected software objects, based on the said conditions of use said securely linked to said protected software objects, where said decide is an autonomous decision, based in part at least, on secure processing of information internal and or external to said secret processing device, and that as long as said the requirements of one or multiple said protected software objects and or said secret processing device are complied with, the user of a said PUCDPS is able to execute and or process one or multiple said protected software object on the same basis as if they were said software object; and or

transfer into said secret processing device and or have transferred any part of one or multiple information that may be necessary to provide any of the functions required by said protected software object; and or

access any information that may be located external to said secret processing device in order to provide any of the functions required by said protected software object; and or

examine said conditions of use said securely linked to said protected software object; and or

determine a response to said conditions of use, and or

respond to said conditions of use; and or

provide one or multiple area of secure memory that is not practical to analyse; and or

provide for partition of secure memory into one or multiple secure system partitions and one or multiple user partitions whereby programs in said system partitions may access said user partitions, however, said user partition may not access said system partition unless authorised, and or any particular said user partition may not access any other said user partition unless authorised; and or

may transfer part or all any one or multiple said protected software object and or any other software objects from unsecure to said secure locations for processing and or transfer any information from said secure location to said unsecure location; and or

may securely decrypt part or all of decrypted parts of said protected software object and or any other encrypted information within said secure locations; and or

may process part or all of one or multiple said protected software object in secrecy, including processing of part or all of that information loaded in encrypted format and decrypted; and or

have the capacity to detect whether part or all of said protected software object have been tampered with; and or

handle the requirements of a large number of different protected software objects that it has not been specifically preconfigured for while in unsecure locations; and or

may perform secret encryption and or secret decryption in a manner that cannot be analysed, and this may be a software and or hardware function; and or

have the capacity to implement in part or whole, one or multiple hardware devices in programmable logic and preferably programmable logic that may be rapidly erased in the event of tampering, and this includes encryption and or decryption functions implemented in part or whole in hardware, and hardware functions implemented in programmable logic may be dynamically programmed by one or multiple protected software object; and or

may use any method to determine that there is an attempt to gain access to secret information within itself, and said attempt may be physical and or logical analysis, and the response may be any action, using any method, including

disabling, temporarily and or permanently, part or all of itself and or invalidating in any way part or all of the secret information that may be stored within secure memory storage devices; and or

may securely store information in encrypted and or clear code format in locations inaccessible to unauthorised parties and or securely store information in encrypted format in locations that may be accessible to unauthorised parties, and may detect tampering with stored information; and or

may have the capacity to securely monitor the usage of said protected software object; and or

may be loaded with information that is any one or multiple units of use, in any secure format, that may be securely stored within said secret processing device and or securely in accessible external locations and said units of use may be used to offset against use of one or multiple said protected software objects as determined by their said conditions of use, said units of use may be adjusted in any way as they are used and may be used to credit various said producer and or said protected software objects and or any other method that can be used to record directly and or indirectly the payments that are due to various producers and any other interested parties;

may securely record the usage of said protected software object and the record may include a secure breakdown of the usage on a producer and or product or any other basis, and said record in part or whole is non-volatile; and or

request and or compel the user of said PUCDPS to provide any necessary reports of usage to said service provider and or to any other location; and or

confirm that said reports that have been received as required; and or

not require modification of the PUCDPS operating system; and or

not require special routines to intercept calls to said system operating system; and or

identify the type of said protected software object and act as required; and or

provide or have access to one or multiple tamperproof, non-volatile source of time and or date; and or

provide or have access to one or multiple tamperproof timers; and or

provide one or multiple method of identifying a particular tamperproof environment that may include the use of an electronic signature; and or

provide one or multiple secret codes and or programs that are unique to a particular secure environment and or that are common across particular groups; and or

provide one or multiple programs, that may be preprogrammed and or transferred as required that use secret information unique to said secret processing device; and or

process multiple said protected software object in a multitasking environment and this may be transparent to said User Controlled Data Processing System; and or

include functions, preferably implemented in reprogrammable secure memory, that may be edited and or modified and or deleted and or expanded and or in any other way changed, in a secure manner and usually transparently to the user of said PUCDPS, enabling externally supplied and appropriately configured said protected software object to adapt the secure processes available to said PUCDPS and create one or multiple applications not currently available to said PUCDPS and or that permits any current application to be dynamically adapted, and said adapt includes dynamically reprogramming various hardware functions implemented in part or whole with reprogrammable logic connections and or dynamically modifying decryption processes; and or

are programs and or data preprogrammed into the device and or transferred in encrypted format and or in clear code that assist any other function that includes the processing of said protected software object; and or

include secure memory that stores various internal system routines and may be loaded with externally supplied objects for decryption and or execution and or any other purpose; and or

may partition secure memory that forms part of said secure and secret processing system into secure system memory and secure user memory, wherein programs within system memory may access those in user memory, however, user programs may not access system memory on an unauthorised basis, furthermore, said user memory may be further partitioned into multiple user partitions, wherein each user partition cannot affect information within other user partitions.

3. A method of distributing software objects according to Claim 1, wherein said not practical may be interpreted as multiple levels of difficulty depending on the requirements and may be too difficult:
for a normal user;
with disassembly of said parts that are not functionally limited,
with attempts at characterising encrypted information in the hope of breaking encryption methods;
with attempts at destroying the package to view the information within.

4. A method of distributing software objects according to Claim1, wherein said Oscar compatible is any functional limitation of part or all of a software object by any method of encryption, usually at a secure location remote to the user, where part or all of the reversal of the encrypted information, by decryption and or any other method, occurs within a secure environment directly and or indirectly attached to a user controlled data processing system such that part or all of the instructions and or data of the software object reconstituted by said reversal are not accessible to analysis by any unauthorised party and the execution of part or all of said instructions and or the processing (using any method) of part or all of said data that is not accessible to analysis by an unauthorised party remains in part or

whole inaccessible to analysis by any unauthorised party. The result is that part at least of the functional limitation placed on a software object is not compromised by the process of using said software object.

5. A method of distributing software objects according to Claim 1, wherein said Groover compatible is any functional limitation of part or all of a software object by deletion of part or all of the information within the software object, usually at a secure location remote to the user, where part or all of the reversal of the deletion, by any other method, occurs within a secure environment directly and or indirectly attached to a UCDPS such that part or all of the instructions and or data of the software object reconstituted by said reversal are not accessible to analysis by any unauthorised party and the execution of part or all of said instructions and or the processing (using any method) of part or all of said data that is not accessible to analysis by an unauthorised party remains in part or whole inaccessible to analysis by any unauthorised party. The result is that part at least of the functional limitation placed on a software object is not compromised by the process of using said software object.

6. A method of distributing software objects according to Claim 2, wherein said determine a response to said conditions may be based on a plurality of information states within and or external to said secret processing device, including the availability of one or multiple said units of measurement to offset against any requirements in said conditions of use, appropriate entry of any data key, compliance with reporting requirements, validation of said conditions of use supplied with said protected software objects against appropriate values stored within said secret processing device.

7. An apparatus for distributing software objects, referenced a secret processing device, that may in part or whole be integrated into the same integrated circuit (and or directly and or indirectly linked) as the system microprocessor of said user controlled data processing system, and preferably does not interfere with the normal functions of said system microprocessor, the secret processing device may also form an integral part of a multiprocessor system microprocessor, part or all of said secret processing device may be integrated into any one or multiple devices external to said system microprocessor and attached directly and or indirectly to said user controlled data processing system;

said secret processing device includes one or multiple secure microprocessors and one or multiple blocks of secure memory storage devices, that may be any type and mix, and may include secure direct memory access controller and other functions as described, wherein said secret processing device may:

securely decrypt and execute and or process instructions and or securely decrypt and process data; and or

securely decrypt and execute and or process instructions and or securely decrypt and process data that complies with part or all of the requirements of reversing functional limitations applied that are said Oscar compatible; and or

reverse any functional limitations applied that are said Groover compatible; and or

reverse part or all any functional limitations applying to said protected software object; and or

may decide to reverse one or multiple said functional limitations applied to one or multiple said protected software objects, based on the said conditions of use said securely linked to said protected software objects, where said decide is an autonomous decision, based in part at least, on secure processing of information internal and or external to said secret processing device, and that as long as said the requirements of one or multiple said protected software objects and or said secret processing device are complied with, the user of a said user controlled data processing system is able to execute and or process one or multiple said protected software object on the same basis as if they were said software object; and or

have the capacity to implement in part or whole, one or multiple hardware devices in programmable logic and preferably programmable logic that may be rapidly erased in the event of tampering, and this includes encryption and or decryption functions implemented in part or whole in hardware, and hardware functions implemented in programmable logic may be dynamically programmed by one or multiple protected software object; and or

transfer into itself and or has transferred any part of one or multiple information that may be necessary to provide any of the functions required by said protected software object; and or

access any information that may be located external to said secret processing device in order to provide any of the functions required by said protected software object; and or

examine the said conditions of use said securely linked to said protected software object; and or

determine a response to said conditions of use; and or

respond to said conditions of use; and or

provide one or multiple area of secure memory that is not practical to analyse; and or

provide for partition of secure memory into one or multiple secure system partitions and one or multiple user partitions whereby programs in said system partitions may access said user partitions, however, said user partition may not access said system partition unless authorised, and or any particular said user partition may not access any other said user partition unless authorised; and or

may transfer part or all any one or multiple said protected software object and or any other software objects from unsecure to said secure locations for processing and or transfer any information from said secure location to said unsecure location; and or

may securely decrypt part or all of decrypted parts of said protected software object and or any other encrypted information within said secure locations; and or

may process part or all of one or multiple said protected software object in secrecy, including processing of part or all of that information loaded in encrypted format and decrypted; and or

have the capacity to detect whether part or all of said protected software object have been tampered with; and or;

may perform secret encryption and or secret decryption in a manner that cannot be analysed, and this may be a software and or hardware function; and or

have the capacity to implement in part or whole, one or multiple hardware devices in programmable logic and preferably programmable logic that may be rapidly erased in the event of tampering, and this includes encryption and or decryption functions implemented in part or whole in hardware, and hardware functions implemented in programmable logic may be dynamically programmed by one or multiple protected software object; and or

may use any method to determine that there is an attempt to gain access to secret information within itself, and said attempt may be physical and or logical analysis, and the response may be any action, using any method, including disabling, temporarily and or permanently, part or all of itself and or invalidating in any way part or all of the secret information that may be stored within secure memory storage devices; and or

may securely store information in encrypted and or clear code format in locations inaccessible to unauthorised parties and or securely store information in encrypted format in locations that may be accessible to unauthorised parties, and may detect tampering with stored information; and or

may have the capacity to securely monitor the usage of said protected software object; and or

may be loaded with information that is any one or multiple units of use, in any secure format, that may be securely stored within said secret processing device and or securely in accessible external locations and said units of use may be used to offset against use of one or multiple said protected software objects as determined by their said conditions of use, said units of use may be adjusted in any way as they are used and may be used to credit various said producer and or said protected software objects and or any other method that can be used to record directly and or indirectly the payments that are due to various producers and any other interested parties;

may securely record the usage of said protected software object and the record may include a secure breakdown of the usage on a producer and or product or any other basis, and said record in part or whole is non-volatile; and or

request and or compel the user of said user controlled data processing system to provide any necessary reports of usage to said service provider and or to any other location; and or

confirm that said reports that have been received as required; and or

not require modification of the PUCDPS operating system; and or

not require special routines to intercept calls to said system operating system; and or

identify the type of said protected software bject and act as required; and or

provide or have access to one or multiple tamperproof, non-volatile source of time and or date; and or

provide or have access to one or multiple tamperproof timers; and or

provide one or multiple method of identifying a particular tamperproof environment that may include the use of an electronic signature; and or

provide one or multiple secret codes and or programs that are unique to a particular secure environment and or that are common across particular groups; and or

provide one or multiple programs, that may be preprogrammed and or transferred as required that use secret information unique to said secret processing device ; and or

process multiple said protected software object in a multitasking environment and this may be transparent to said User Controlled Data Processing System; and or

include functions, preferably implemented in reprogrammable secure memory, that may be edited and or modified and or deleted and or expanded and or in any other way changed, in a secure manner and usually transparently to the user of said PUCDPS, enabling externally supplied and appropriately configured said protected software object to adapt the secure processes available to said PUCDPS and create one or multiple applications not currently available to said PUCDPS and or that permits any current application to be dynamically adapted, and said adapt includes dynamically reprogramming various hardware functions implemented in part or whole with reprogrammable logic connections and or dynamically modifying decryption processes; and or

are programs and or data preprogrammed into the device and or transferred in encrypted format and or in clear code that assist any other function that includes the processing of said protected software object; and or

include secure memory that stores various internal system routines and may be loaded with externally supplied objects for decryption and or execution and or any other purpose.

8. A method of distributing software objects according to Claim 7, wherein said determine a response to said conditions may be based on a plurality of information states within and or external to said secret processing device, including the availability of one or multiple said units of measurement to offset against any requirements in said conditions of use, appropriate entry of any data key, compliance with reporting requirements, validation of said

conditions of use supplied with said protected software objects against appropriate values stored within said secret processing device.

9. An apparatus for distributing software objects according to Claim 7, wherein said Oscar compatible is any functional limitation of part or all of a software object by any method of encryption, usually at a secure location remote to the user, where part or all of the reversal of the encrypted information, by decryption and or any other method, occurs within a secure environment directly and or indirectly attached to a user controlled data processing system such that part or all of the instructions and or data of the software object reconstituted by said reversal are not accessible to analysis by any unauthorised party and the execution of part or all of said instructions and or the processing (using any method) of part or all of said data that is not accessible to analysis by an unauthorised party remains in part or whole inaccessible to analysis by any unauthorised party. The result is that part at least of the functional limitation placed on a software object is not compromised by the process of using said software object.

10. An apparatus for distributing software objects according to Claim 7, wherein said Groover compatible is any functional limitation of part or all of a software object by deletion of part or all of the information within the software object, usually at a secure location remote to the user, where part or all of the reversal of the deletion, by any other method, occurs within a secure environment directly and or indirectly attached to user controlled data processing system such that part or all of the instructions and or data of the software object reconstituted by said reversal are not accessible to analysis by any unauthorised party and the execution of part or all of said instructions and or the processing (using any method) of part or all of said data that is not accessible to analysis by an unauthorised party remains in part or whole inaccessible to analysis by any unauthorised party. The result is that part at least of the functional limitation placed on a software object is not compromised by the process of using said software object.

11.An apparatus for distributing software objects according to Claim 7, wherein said protected software object is a software object that has been reversibly functionally limited to be reversed in part or whole by functions provided by said secret processing device.

12 An apparatus for distributing software objects according to Claim 7, wherein said conditions of use may be a plurality of conditions securely linked to said protected software object that are extracted in part or whole by said secret processing device and used to determine whether to reverse the said functional limitations applied to one or multiple said protected software object.

13 A method of securely protecting and distributing software objects substantially as herinbefore described with reference to the drawings.

14. An apparatus for distributing software objects substantially as herinbefore described with reference to the drawings.

15. The steps, features, compositions and compounds disclosed herein or referred to or indicated in the specification and/or claims of this application, individually or collectively, and any and all combinations of any two or more of said steps or features.

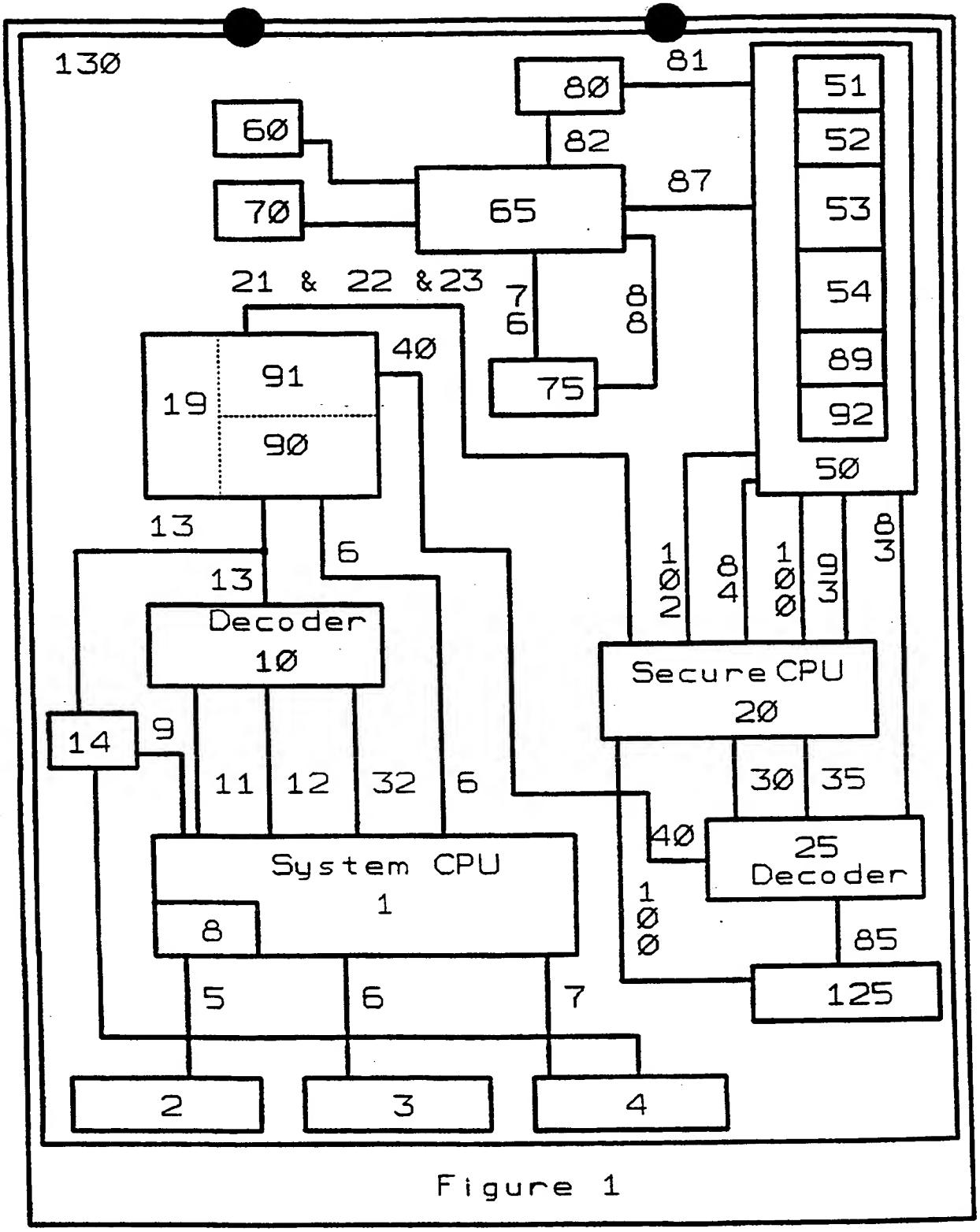

130
80
81
51
52
53
54
89
92
50
60
82
70
65
87
21 & 22 &23
76
88
40
91
19
90
75
13
6
13
Decoder
10
14
9
11
12
32
6
102
84
100
93
83
Secure CPU
20
30
35
40
25
Decoder
System CPU
1
8
100
85
125
5
6
7
2
3
4

Figure 1

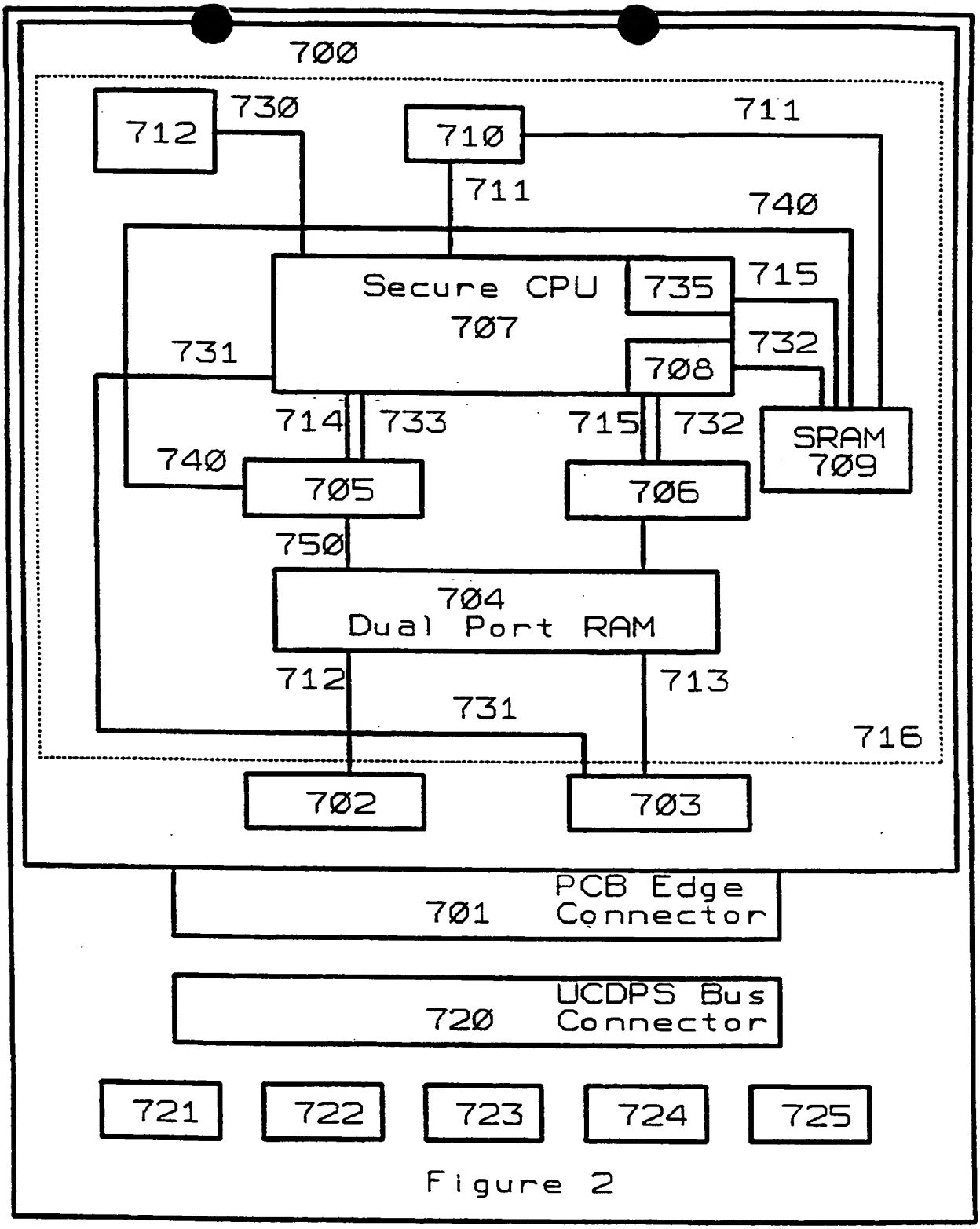

700
712
730
710
711
711
740
Secure CPU
707
735
715
731
708
732
714
733
715
732
SRAM
709
740
705
706
750
704
Dual Port RAM
712
713
731
716
702
703
701
PCB Edge
Connector
720
UCDPS Bus
Connector
721
722
723
724
725

Figure 2

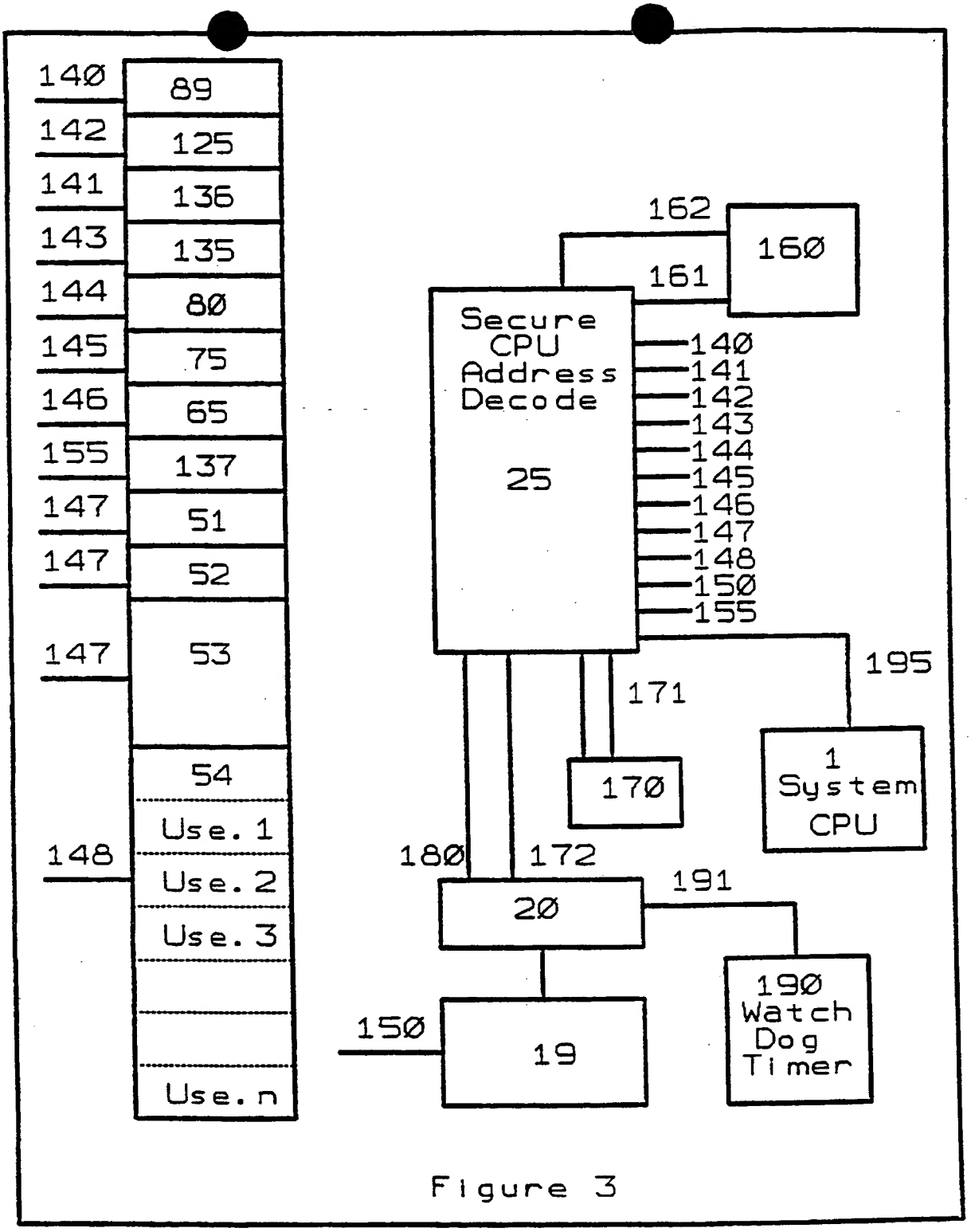

140
89
142
125
141
136
143
135
144
80
145
75
146
65
155
137
147
51
147
52
147
53
54
Use. 1
148
Use. 2
Use. 3
Use. n
162
160
161
Secure CPU Address Decode
25
140
141
142
143
144
145
146
147
148
150
155
195
171
170
1 System CPU
180
172
191
20
190 Watch Dog Timer
150
19

Figure 3

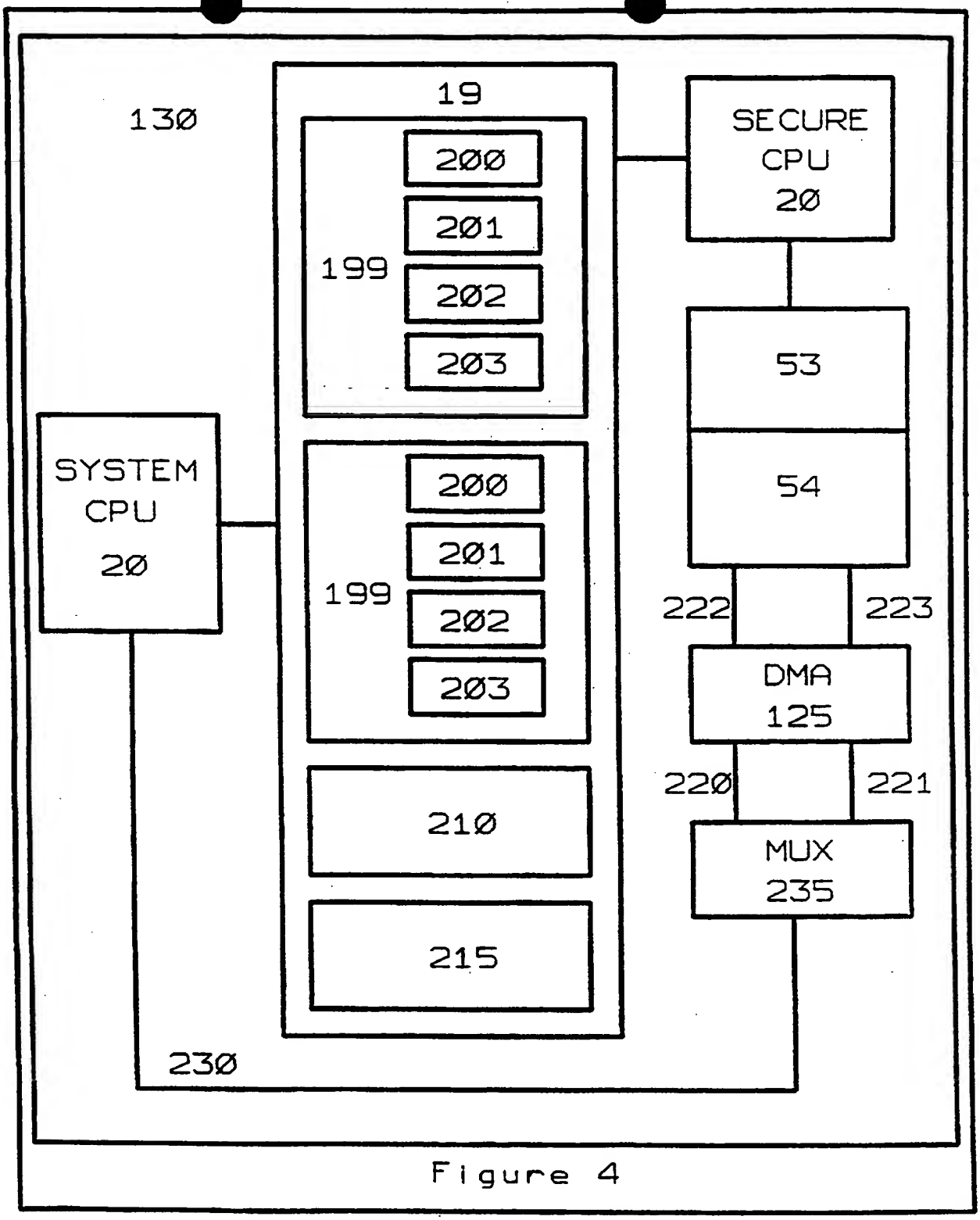

130
19
200
201
199
202
203
SECURE CPU 20
53
54
SYSTEM CPU 20
200
201
199
202
203
222
223
DMA 125
220
221
MUX 235
210
215
230

Figure 4

INTERNATIONAL SEARCH REPORT

International Application No.
PCT/AU 97/00010

A. CLASSIFICATION OF SUBJECT MATTER

Int Cl[6]: G06F 12/14

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC: G06F 12/14

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
AU: IPC as above

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	WO-A-9522796 (INFOSAFE SYSTEMS INC) 24 August 1995 See whole document	1-15
X	WO-A-9321581 (SECURE COMPUTING CORORATION) 28 October 1993 See page 10 line 18 to page 11 line 35 and page 18 line 35 to page 19 line 7	1-15
X	EP-A2-561685 (FUJITSU LIMITED) 22 September 1993; See column 4 lines 15-25	1-15

[X] Further documents are listed in the continuation of Box C

[X] See patent family annex

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
7 April 1997	16 APR 1997
Name and mailing address of the ISA/AU AUSTRALIAN INDUSTRIAL PROPERTY ORGANISATION PO BOX 200 WODEN ACT 2606 AUSTRALIA Facsimile No.: (06) 285 3929	Authorized officer Michael Hardy Telephone No.: (06) 283 2547

INTERNATIONAL SEARCH REPORT

International Application No.
PCT/AU 97/00010

C (Continuation) DOCUMENTS C NSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	WO-A-9214209 (TOVEN TECHNOLOGIES INC) 20 August 1992; See whole document	1-15
X	WO-A-9013865 (SOFTEL INC) 15 November 1990; See whole document	1-15
X	EP-A2-266748 (INTERNATIONAL BUSINESS MACHINES CORPORATION) 15 May 1988 See column 4 line 28 to column 15 line 16	1-15

INTERNATIONAL SEARCH REPORT

Information on patent family members

International Application No.
PCT/AU 97/00010

This Annex lists the known "A" publication level patent family members relating to the patent documents cited in the above-mentioned international search report. The Australian Patent Office is in no way liable for these particulars which are merely given for the purpose of information.

Patent Document Cited in Search Report		Patent Family Member					
WO	9522796	AU-A1	19236/95	US-A	5394469		
WO	9321581	AU-A1	42847/93	AU-B2	667925	AU-A1	50811/96
		EP-A1	636259	EP-A2	737907	JP-T2	7505970
		US-A	5276735	US-A	5499297		
EP	561685	JP-A2	5257816	US-A	5392351	US-A	5555304
WO	9214209	AU-A1	12009/92	CA-A	2035697	US-A	5325430
WO	9013865	AT-E	143511	AU-A1	56464/90	AU-B2	641397
		CA-A	2053261	CN-A	1048271	DE-C	69028705
		EP-A1	478571	JP-T2	4504794	US-A	5388211
		US-A	5497479				
EP	266748	DE-C	3751047	JP-A2	63127334	US-A	5109413

END OF ANNEX

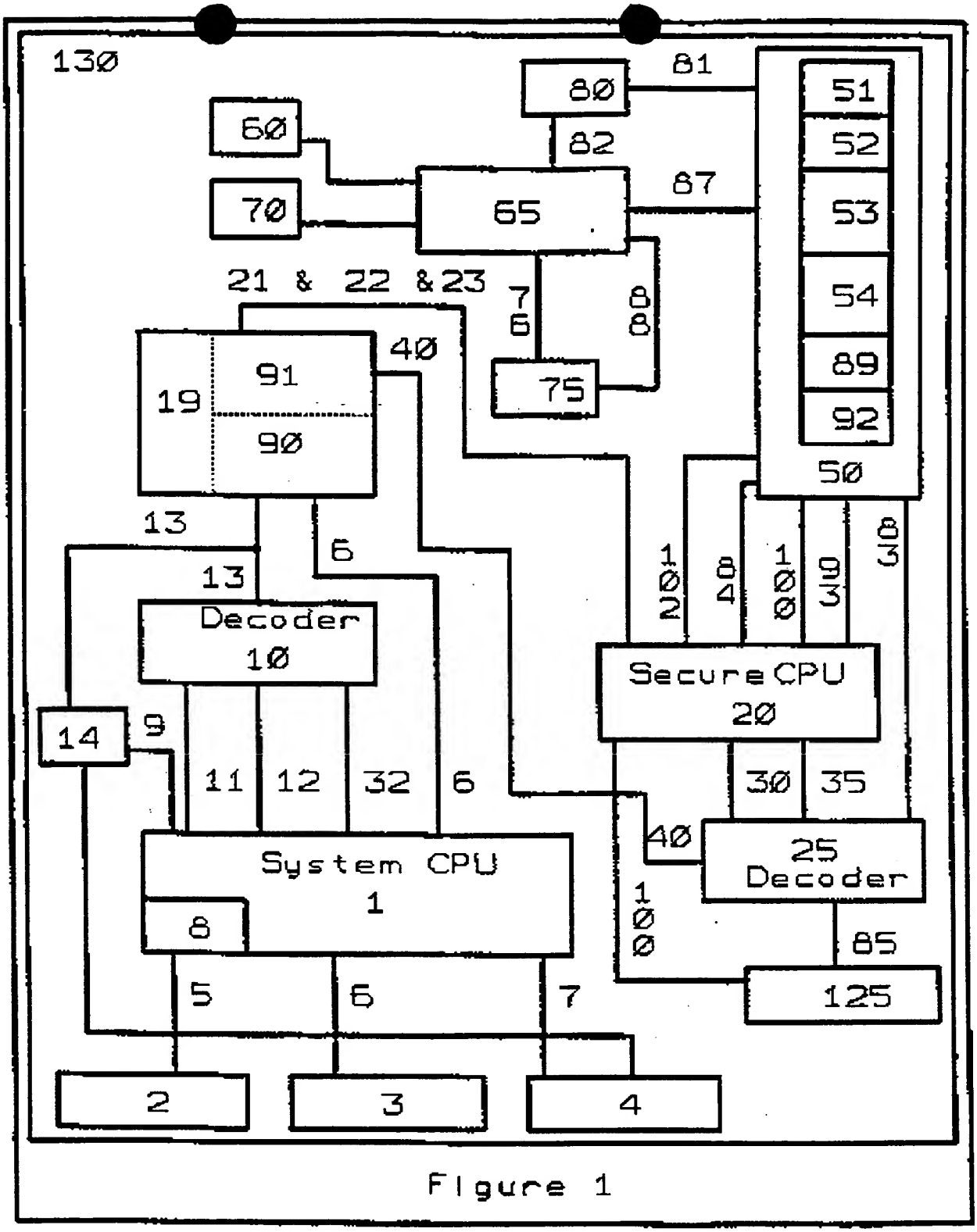

130
80
81
51
52
53
54
89
92
50
60
70
82
65
87
21 & 22 & 23
76
88
75
40
91
90
19
13
13
Decoder
10
6
120
48
100
36
38
Secure CPU
20
14
9
11
12
32
6
30
35
40
25
Decoder
System CPU
1
8
100
85
125
5
6
7
2
3
4

Figure 1

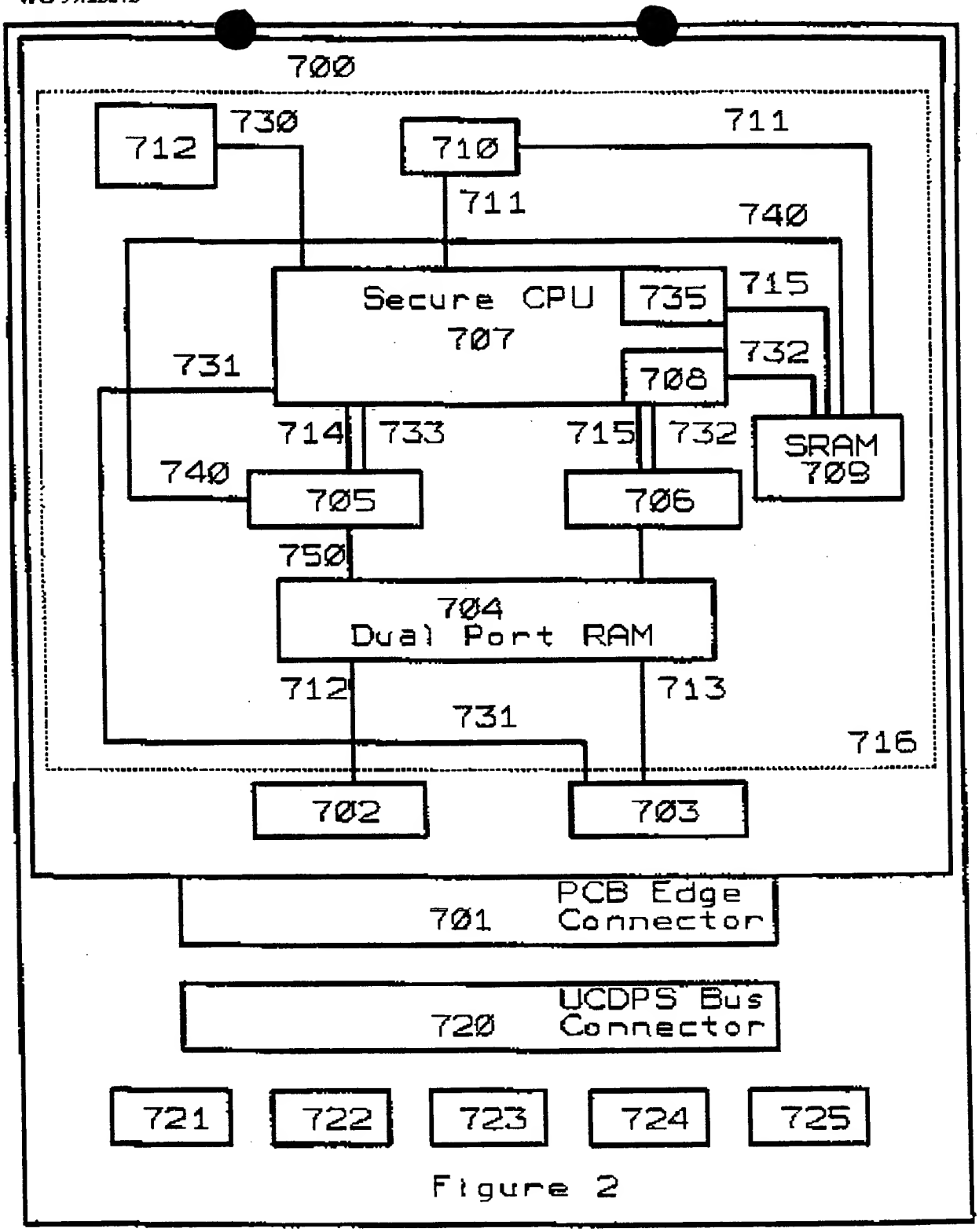

700
712
730
710
711
711
740
Secure CPU
707
735
715
731
708
732
714
733
715
732
SRAM
709
740
705
706
750
704
Dual Port RAM
712
713
731
716
702
703
701
PCB Edge
Connector
720
UCDPS Bus
Connector
721
722
723
724
725

Figure 2

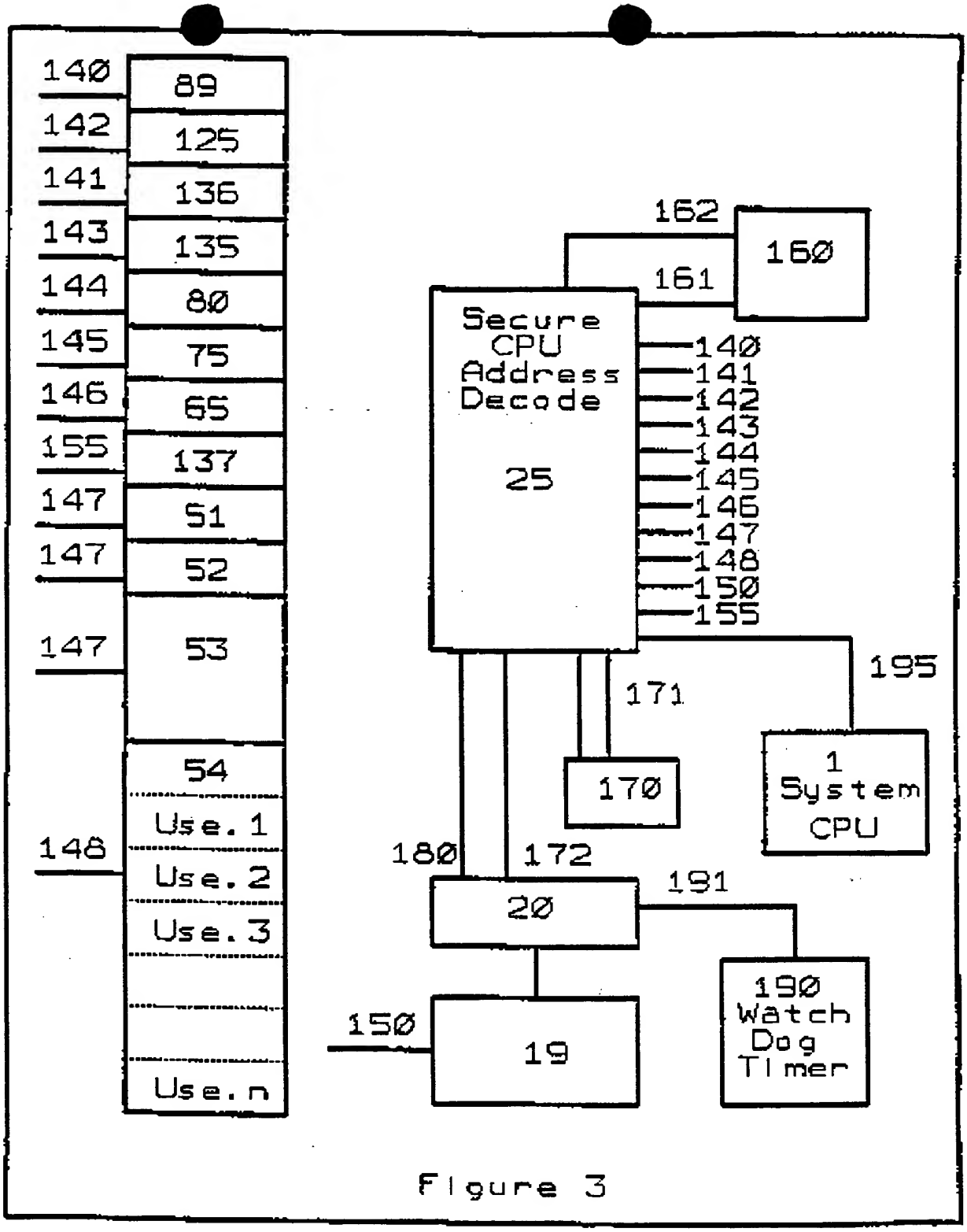

140
89
142
125
141
136
143
135
144
80
145
75
146
65
155
137
147
51
147
52
147
53
54
Use. 1
148
Use. 2
Use. 3
Use. n
162
160
161
Secure CPU Address Decode
25
140
141
142
143
144
145
146
147
148
150
155
195
171
170
1 System CPU
180
172
191
20
190 Watch Dog Timer
150
19

Figure 3

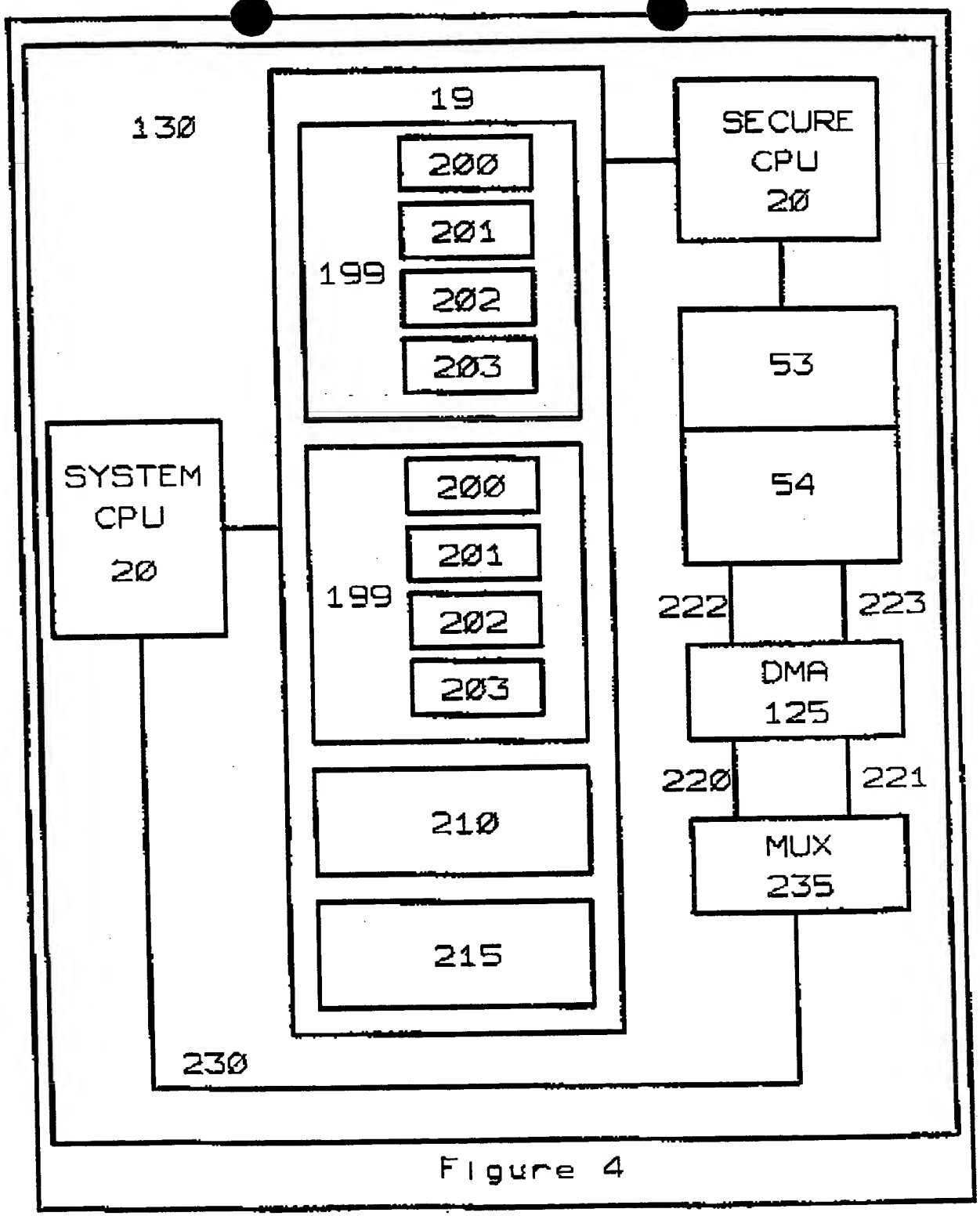

130
19
200
201
199
202
203
SECURE CPU 20
53
54
SYSTEM CPU 20
200
201
199
202
203
222
223
DMA 125
220
221
MUX 235
210
215
230

Figure 4